Title Author	HESEA-ACE 11/00/84	Source
ATR-8000 Word Machine		AACE
Bill Rammer	.....	09/00/84 n
Atari DOS Hidden Functions		ORNJUCE
Oregon ACE	.....	08/00/84 n
AtariWriter Files over Modem		C-Squad
Unknown	.....	08/00/84 n
Basic Modification		AARI
Don Murphy	.....	09/00/84 n
Cartridge Switches		C-Squad
SIG-ATARI	.....	08/00/84 n
Compuserve		SLO-POKES
T. Wiggins	.....	10/00/84 n
DIF Files		AARE
David Fuller	.....	09/00/84 n
Entering Machine Programs into Strings		ABACUS
I.F. Carlstrom	.....	09/00/84 n
Graphics Tablet Drawing Tips		PACE
Marion Delahan	.....	07/00/84 n
Internal Keycodes		TCC
Unknown	.....	11/00/84 n
Interrupt Structure & the Serial Port		MILATARI
Donald Wilcox	.....	10/00/84 n
Joystick Ports		AACE
Rick Detlefsen	.....	09/00/84 n
Null Modem Cable		PACE
Frank Magle	.....	07/00/84 n
Parts for Atari		AARI
Bill Dwyer	.....	08/00/84 n
Parts for Atari		WAND
Unknown	.....	11/00/84 n
Parts for Atari		STARFLEET
B & C Computer Vision	.....	09/00/84 n
Peeks & Pokes		SLO-POKES
Dave Grose	.....	09/00/84 n
* Program Documentation		WAND
Rolly Herman	.....	11/00/84 n
Programming Languages, A Guide to		JACG
Arthur Leyenberger	.....	10/00/84 n
* Prowriter Printer Dip Switch Settings		ORNJUCE
Computah	.....	08/00/84 n
SYNCOM		PACE
Gordon Andersen	.....	09/00/84 n
SYNCRON		PACE
Gordon Andersen	.....	09/00/84 n
SYNFILE+		ORNJUCE
Greg Beauton	.....	08/00/84 n
SYNSTOCK		PACE
Gordon Andersen	.....	07/00/84 n
Speeding up Atari BASIC		TCC
Tom Disque	.....	10/00/84 n
String Control with Control Characters		WLAAUG
Gerry Wick	.....	10/00/84 n
USR Function		CURRENT NOTES
Ernie Rice	.....	11/00/83 bn
VISICALC Printer Control (Bit by Bit)		PACE
Steve Monn	.....	07/00/84 n
Vertical Bland Interrupts		ORNJUCE
Harjit Singh	.....	09/00/84 n
Write Enable-Disable Switches		C-Squad
Paul Surowiec	.....	09/00/84 n

ATR-8000 SIG

AN ATR-8000 USER - SPECIAL INTEREST GROUP
AN AACE SIG

AACE ATR8000 NEWSLETTER for September, 1984

This month we are featuring contributions from members:

Review of "WORD MACHINE" CP/M word processing program

By Bill Rammer...Great Falls, Montana

For you ATR-8000 owners that are looking for a rather good CP/M word processing program at a very nominal cost of $37.50..."WORD MACHINE"..., produced by GB Associates, might be what you need. I have been using Letter Perfect for the past two years and have liked it very well but did not care for the limitations of 40-column monitor and limited disk space storage. I tried "WORD-STAR" and found it much too cumbersome for home or home-business use. (I don't like to answer seven questions every time I want to print something nor do I like to save document before I can print it...and on & on.)

WM is sold on a SD/SS disk configured to TRS-80 MOD III. Diskdef selection for TRS-MOD I will read this without any problems. I would have opted for the 80-column Letter Perfect but I was burned once by the Austin-Franklin board and am reluctant to add expensive hardware just to get 80 columns. WM when used with the inexpensive SWP 80-column disk or the DT-80 ROM make a very inexpensive word processing system especially when there is 380K of storage on one of our disks.

The WM is user reconfigurable for the Centronics printers (Radio Shack Lineprinters), the Epson MX series, all Daisywheel printers and most other printers with a parallel interface. When configured for the Centronics or Epson, it supports all available print styles. The Epson version also allows changing line spacing, all under program control with the text.

The WM offers these features:

- Text input from keyboard or from disk
- Fully automatic wrap-around
- Global search and replace
- Complete editing using Microsoft Basic commands.
- Insert or delete any number of lines
- Block move lines from one part of text to another.
- Right justify the lines if desired.
- Format text to longer or shorter line width than originally typed.
- Save text on DD/DS disk if desired.
- Get a listing of all text files on disk.
- Print text with choice of line numbers, page numbers, offset for left-hand binding, set top margin and left hand margin, choice of number of lines per page, single page or fanfold printing.

I found it a little strange at first to edit using Microsoft Basic commands but it grows on a person shortly and becomes very easy to use. The documentation provided is brief but clear. The manufacturer continues to make improvements to the program and will update for owners if they send their original program disk and mailing costs. There are probably better word processor programs out there, but for $37.50? If you have to do layout work, in 80 columns on the monitor, this program will do it.

A minor nuisance with this program is that you cannot right-justify a single line of text such as the date group on a letterhead. Another problem is when the document is saved, the printing formats such as margin spacings and lines per page etc. are lost if reloaded from disk storage. This requires you to remember how the margins etc. were formatted before printing. On the positive side, a line of text can be centered, however. In addition this program will print as little as one line, a paragraph or the whole document. In the edit mode, the individual lines of the document are numbered.

There are many more features too minor to mention. This program is so much faster and easier to use than Word-Star that it is a pleasure to have. The program is produced by:

GB ASSOCIATES
P.O. BOX 3322
GRANADA HILLS, CA 91344

I am...Bill Rammer
406-727-5221
1109 19th Ave. SW
Great Falls, MT 59404

HEATHCLIFF By Geo. Gately

"Stay away from my home computer!"

ORNS/OUG AC AOC
AUG 84

ATARI DOS....

Hidden Functions

Reprinted from 'Oregon ACE'

FUNCTION ONE

What happens when you can't find your Basic Cartridge, and you really want to type in that new program listing you found? Well, don't worry because there is a way to type in Basic and maybe even another language through Atari DOS.

In order to do this, first, go to DOS and when the menu appears, type a 'O' and RETURN. You will then be prompted with COPY FROM, TO? Enter 'E,D:filename' and RETURN. You will hear the disk drive go on and open a file to the filename entered. The drive light will go out soon after it goes on leaving the cursor one line below the last line typed.

You can now type in a Basic program just as though you were in Basic. At the end of each line hit RETURN just as in Basic. The only limit to the length of the program you type in is the memory of your computer. Be careful! Most mistakes because the only way you can correct them is by typing in the line again. So proofread carefully before you hit the RETURN key.

I think you will find this an interesting function of Atari DOS. You are not limited to Basic programs. You may also type in data for programs or a message for a friend. To stop this function hit the CTRL key an the '3' at the same time and your program will be LISTED out to the disk. Do not refrain from using inverse characters, lower case or control characters. The Editor will accept anything you type into it. To read back messages, simply hit 'O' and 'filename,' and your message will be loaded into memory and be displayed. To stop the scrolling hit CTRL 1; hit it again to continue. To load a program typed in through DOS, simply go to Basic and 'ENTER'D:filename'.

FUNCTION TWO

You are in DOS and you want to reboot the system or just return to Basic. One way is to hit the 'B' function of DOS, which is 'RUN CARTRIDGE' will exit to the cartridge you have in the left slot. Another way is through the 'M' function of DOS 'RUN AT ADDRESS'. Select this function, enter E486 or E477 when the computer asks 'RUN FROM WHAT ADDRESS' and hit RETURN. The later will do a 'COLDSTART', the same way as if you shut the machine off and turn it back on again. When you do this, all memory locations will be cleared and the computer will execute the cartridge if you have one in. If you want to do a 'WARMSTART', instead of intering E486, enter E484 of E474 and the computer will perform a 'WARMSTART', the same as hitting the 'SYSTEM RESET' and execute the left cartridge.

Frank and Earnest

DIP SWITCH SETTINGS....

for the

Prowriter Printer

Reprinted from the COMPUTAH,
Salt Lake City, UT

ACAOC
AUG 84

(3)

It seems that when you try to print something in inverse characters with the Prowriter printer it comes out Japanesse. It also seems that when done with the ATARI File Manager that everything come out looking that way

You can reset to the dip switch settings below and then it behaves itself nicely. The inverse characters come out in regular print style, but that's better than it was in Japanesse

'O' IS ON, 'X' IS OFF ON ALL SWITCHES.

SW-1	SW-2
1 2 3 4 5 6 7 8	1 2 3 4 5 6 7 8
O X O O O X X X	X O O O O X X O

That should do the trick, if you know of any other improvments or necessary changes please let me know and we will publish them

EXPOSE YOURSELF
TO A COMPUTER !!

C-SQUAD 8-84

(4)

TIMELY TIPS

Sending ATARIWRITER files over the modem

The printer formatting codes at the top of the page on an ATARIWRITER file present a serious problem when sending the file over a modem to a non-ATARI BBS or a non-ATARI computer. Unless you erase these codes first, they are sent along with the file. The second format code which is (CNTRL-D) is in ASCII and means end of transmission. The computer on the receiving end will get the code and hang up the phone.

To take those format codes off your file, make your printout and go to the top of the file. The cursor's home position is in the upper left hand corner below the codes. Although the codes are above the home position, the cursor will go there.

Move the cursor up to that line and hit (SHIFT DELETE BACK SPACE). This will zap the codes.

The files are not able to be sent over the modem to any computer without problems. The other computer will not be able to read your ATARIWRITER'S (CONTROL P) and (RETURN) so your file will come through as one long paragraph. Still, this is better than not coming through at all.

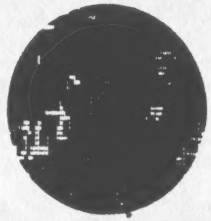
Loading Binary Load Files on Your New XL from cassette

When loading binary load files on your ATARI XL from cassette, don't forget to hold down both the START and OPTION keys to lock out BASIC.

Printing a Disk Directory from DOS

If you have ever wanted to print out a list of the files on a disk before you exit to BASIC, all you have to do is press A (carriage return) and when it asks for drive number and destination, type ,P: for a printout of the files on drive one or type D2,P: for a printout of the files on drive two.

```
**************************************
*                                    *
*        PROGRAM RATINGS             *
*            from                    *
*        The Gamemasters             *
*                                    *
*                          12345678901*
*SUMMER GAMES              ********* *
*BEACH HEAD                ******** *
*HUMPTY DUMP               ******** *
*MONTEZUMA'S REVENGE       ********* *
*CARNIVAL MASSACRE         ******* *
*DRAGON'S KEEP             ***** *
*TROLL'S TALE              ***** *
*SPACE KNIGHTS             ****** *
*COLLOSAL CAVE             ******* *
*FINAL EXAM                ***** *
*EVEREST EXPLORER          **** *
*SCANALYZER                ******** *
*LOGO                      **********
*DRAPER PASCAL             ********* *
*MICROSOFT BASIC           ******* *
*                                    *
**************************************
```



Cursor out of range!

>>>> HARDWARE MODIFICATIONS <<<<

ATARI 9-87

A Basic Modification

by Don Murphy

If your an owner of an atari 800, you probably noticed that there are two cartridge slots but 99.9 percent of the cartridges are left slot cartridges. You have also noticed that continually pulling the basic cartridge in and out of the slot wears out the gold contacts and will eventually render both cartridge and slot unusable. But by making the following modifications, Basic can be kept in the right slot almost permentally and freeing up the left slot of wear and tear. Basic can be accessed by means of a switch or two. This modification will prolong the life of both cartridge and slot.

First remove all covers and shields from your atari 800 to fully expose the motherboard. Remove any ram boards and the CPU board from the mother board and set them aside. Next place the mother board on a flat working suface with the solder side facing up. Locate the area where the cartridge slots have been soldered in. Refere to figuire 1 while the procedure is discribed and remember that the left cartridge slot is now on the right because its upsidedown.

Break the leds running to pins 1,A,12,13, and 14 to the right slot. To break the leads do it with an exacto knife at a convienent point by scratching the board. On the left slot break leds 12,13 and 14. Run a wire from pin 1 on the right slot to pin 1 on the left slot also run a wire from pin A on the right to pin A on the left slot. Refere to figuire 2 while the wiring of the switches is described

Figuire 1

break the leads to the x-ed pins and run the two wires shown

Cut 9 pieces of wire about 1 1/2 to 2 feet long and lable them as follows:

1)12L	2)12R	3)12C
4)13L	5)13R	6)13C
7)14L	8)14R	9)14C

Solder them in place by their number and letter. For example 12L should be soldered right on to pin 12 of the left slot, 12R should be soldered right onto pin 12 of the right slot, 12C should be soldered to the lead that used to run to pin 12 of the left slot- do not solder it to the pin but to the lead. Do the rest of the wires in the same manner and watch to make sure solder contacts only what its suppose to- avoid solder bridges.

figuire 2:

Put the Ram boards and CPU board back on the motherboard and reassemble the shielded portion of your computer. Cut a notch on the bottom shield to make room for the 9 wires that eventually will be hooked to the switches. After all the shields are in place reassemble the rest of the computer leaving the top off. With the computer facing up drill holes for the switches on the left side of the plastic consol near the front. Make sure the holes are on the bottom and not the top piece. Once the switches are mounted run the 9 wires as shown:

12L=>1	12R=>3	12C=>2
13L=>7	13R=>9	13C=>8
14L=>4	14R=>6	14C=>5

The switches should be DPDT switches and you'll need two of them even though only 1 1/2 is used. They are switches available to do it with only one switch all you need is a switch that will throw 3 leads in two direction. Now you can reassemble the computer and try it out. Put basic in the right cartridge slot and what you what in the left slot or leave the left empty to boot DOS. Throwing the switches in the same direction either way will either turn on basic or boot what ever is in the left slot.

If it doesn't work retrace the lines and make sure they are connected properly. Watch out for solder bridges and make sure both switches are in the same direction. Most importantly power down before switching the switches failure to do so will result in the calapse of the system. IF you have checked everything and it still doesn't work i can be reached at my home phone most days after 4 pm at (617) 532-4297 ask for Don. One last thing this modification basically removes the right cartridge from the system if you have any valuable right cartridges don't do this modification! If it workes you can now leave basic in the right slot and use the left slot for anyother cartridge!!!!!

C-SQUAD
8-84

CARTRIDGE SWITCHES

When you plug a ROM cartridge into the left slot of your Atari 800 computer, you disable the top 8K of RAM. This is done by disabling one input of an OR gate (Z102B) that normally passes the address lines A15 and A14, decoded by chip Z101 to be S5, to the RAM slots. The S5 signal is wired to the left cartridge slot, pin 12, to enable the ROM chips in the cartridge. Pin 14 of the cartridge is connected inside to the +5 volt line. When the cartridge is inserted into the left slot this +5 volts is then connected to the Z102B OR gate to disable the S5 signal to the RAM slots. The S5 signal is the address for 40K to 48K of RAM.

The right cartridge does the same thing, except it uses A13 and A13, decoded by Z101 as S4, for its enable line. Pin 14, the +5 volt signal, of the right cartridge, disables S4 to the RAM slots with OR gate Z102A. The S4 signal is the address for 32K to 40K of the RAM.

If, while the cartridge is inserted, the +5 volt signal to the OR gate could be opened, the RAM would then be enabled. If the S5 line to the left cartridge is also opened, the cartridge chips data output would be tri-stated. (Tri-stated is a third binary output state for digital chips. It is a high impedance state that electronically disconects the chip from the data buss).

Since the address lines, A0 to A12, are inputs, they can be left on the buss. By using a switch, mounted on the case, the programmer can select if RAM or the cartridge ROM is on the data buss. A second switch will do the same for the right cartridge ROM.

If the S5 enable line, normally going to the left slot, is switched to the right slot, the right cartridge will be addressed as a left cartridge. You also have to switch the +5 volt signal from the right cartridge to the Z102B gate to turn off the 40K to 48K RAM.

If your computer is under warranty, don't modify it!

The parts needed are two miniature toggle switches. Both are double pole, double throw, one is a two position (on-on), and the other is a three posistion (on-off-on) switch. A two foot length of eight conductor ribbon cable (Unless you planed ahead and put in a ten conductor ribbon cable with the reset modification last time) and 10-12 inch lengths of small insulated wire.

Once you have the parts and tools, proceed to disemble the computer to the mother board. Don't forget the CMOS handling caution!

The first step is to drill a hole near the center of the board for two small wires to pass through from the top to the bottom of the board. Be careful not to drill near or through any circuit runs. Hold the mother board up to a strong light to be able to see the runs on the bottom of the board and mark the location with a felt pen.

Cut the five runs by making two cuts across the run about 1/16 inch apart, then heat the 1/16 piece with a soldering iron until it lifts off the board.

1) From R109 to left cartridge pin 14.

2) From feed through to left cartridge pin 12.

3) From Z102 pin 4 to left cartridge pin A.

4) From Z102 pin 5 to feed through.

5) From Z101 pin 5 to feed through.

Next, run an insulated jumper from Z102 pin 5 to Z101 pin 5. Scrape the solder mask from the run just above where you made the cuts and solder the jumper to the run. Be careful with the soldering iron, remember how easy it was to remove the 1/16 inch cut out piece? Check your work carefully as you go to be sure the wires are soldered well and there are no solder bridges between runs.

Next, mount the connector in the lower right corner of the mother board, if you did not do so last time.

Now run eight wires (small, solid, insulated telephone wire) from the cut circuit runs to the connector as follows.

1) From R108 on the top through the hole to connector pin 5 on the bottom.

2) From Z101 pin 5 on the top through the hole to connector pin 6.

3) From left cartridge connector pin 1 to connector pin 7.

4) From left cartridge connector pin A to connector pin 8.

5) From left cartridge connector pin 12 to connector pin 3.

6) From left cartridge connector pin 14 to connector pin 4.

7) From R109 solder pad to connector pin 1.

8) From Z102 pin 2 (at the feed through) to pin 2.

(continued from page 21)

Now drill two holes and mount the cartridge select switches on the left of the case top. Be sure the center off switch is to the left when viewed from the top.

The eight wires from the connector plug will now be connected. The order of the wires to the connector plug is not specified. Above each wire write in the color of the wire you have coming from the connector plug.

Solder the two jumpers from the right switch to the left switch. Be sure they are installed as shown. Use heat shrink tubing on all switch connections to be sure there are no stray wires to cause shorts. Connect the eight wires from the connector plug to the cartridge select switches as shown. Then trace each wire to be sure they are connected properly!

Now is the time to check all of your work carefully to be sure there are no shorts or solder bridges or frayed wires any where, and that all connections are proper!

Reassemble your computer and cable it to your system. Install the BASIC cartridge in the left slot and set both switches ON (up). Leave the disk drive off and power up. You should see the familiar READY prompt on the screen. If you don't then check that both switches are ON (up). If they are, then you have a mistake in your wiring. You will have to disemble the computer and check the wiring again. Be sure to check which pin you used as #1 on the new connector.

If at first you get the READY prompt then flip the left switch to the OFF (center) position and do a COLD reset. You should now have the memo pad title. Flip the left switch down (RIGHT cartridge position) and do a COLD reset. You should still have the memo pad title.

Flip both switches ON (up), and do a COLD reset. You should have the BASIC ready prompt. In direct mode execute the following command: ? FRE(0). The number you see printed is the amount of free RAM you have. Make a note of this number then install another cartridge in the RIGHT slot. With both switches ON (up) you should get the BASIC ready prompt. Execute the ? FRE(0) command again and compare the number printed on the screen with the number you got before. It should be 8192 less. This is because the cartridge in the right slot deselected 8K of RAM. Flip the RIGHT switch OFF and do a COLD reset then execute the command ? FRE(0) again. You should get the original number on the screen, because the right cartridge has been electronically removed from the buss.

Flip the LEFT switch OFF and do a COLD reset. You should now have the Memo pad title. Flip the LEFT switch to RIGHT (all the way down) and do a COLD reset and you should see a screen appropriate to the cartridge you have in the RIGHT slot.

Flip the LEFT switch ON and the RIGHT switch OFF and turn on the disk drive. When the busy light goes out insert a diskette with DOS on it and do a COLD reset. The screen should have the BASIC READY prompt, or what ever is appropriate for the software on your diskette. Flip the LEFT switch OFF and do a COLD reset. The disk should reboot and come up with the DOS menu.

If you have a cartridge to disk copier, you can forget jamming the cover switch and inserting the cartridge to be copied in the right slot with the power on. Just insert the cartridge in the right slot and flip the RIGHT cartridge switch OFF and close the cover. When the software instructs you to insert the cartridge, just flip the RIGHT switch ON.

This article is based on information found on SIG*ATARI on Compuserve.

AMS AND THE IL MACHINES

There is now a version of the Advanced Music System which will run on the new IL line of computers. This set of three programs is being added to the club library along with a public domain IL fix designed to take over where the translator leaves off.

FOR YOUR INFORMATION

To make the programs in the newsletter easier to read, they are no longer printed in condensed mode.

All of the artwork in this issue was dumped to printer with HUMPTY DUMP, a new screen dump program for the ATARI home computers.

COMPUSERVE
A Tour and Discussion

If you read my notes last month on THE SOURCE, you know I didn't care for it. If you didn't read it, there's no excuse! Send me $29.95 for a back issue.

After seeing that several of the magazines listed their authors names and COMPUSERVE ID'S, I thought it might be time to check the service out. Most computer stores and SEARS offer the starter pack for the service. It costs from $25.00 to $35.00. The pack includes 5 hours of log-on time and a small notebook on the service.

Before I go any further, let explain that these services run on large mainframes and the sheer size of them is awe inspiring. It can take you hours just to figure out how some of the functions work. While there are many different services, games, forums, etc., it seems that a different set of control codes is necessary to operate each one. For instance, there is a manual ($3.95) just for the computer Special Interest Groups (SIG's)! Another manual just for the games (Dec-Wars, Mega-Wars, etc.). Even though this can get confusing, there is always, and I mean always a help or instructions function handy no matter where you are on the service.

Well skeptical Tom plunges in armed with his small manual, password and 5 hours of time. Guess what? After my five hours had been used up, my sole accomplishment was figuring out how to read messages on the Atari SIG. I spent some time giving myself an overview of the various functions and looking at stuff too. So I had to continue my membership by giving them my credit card number. Since I was still having a bit of trouble with some of the functions, I ordered the big manual from the on-line ordering desk. It arrived in about 2 weeks. Surprise! It wasn't much more detailed than the small manual I already had...in fact, about 75% was repeat information. Now here's a dumb question. If the only way to sign-on is by purchasing the Starter Pack with manual, why does the $14.95 manual you order from them duplicate that much information? I was beginning to smell a fish, and this ain't Denmark.....

Before I dissect this fish, I must admit that COMPUSERVE is a great service. It offers tremendous games on-line, a superb Atari SIG, encyclopedias, CB simulators, Airline Guides, Color hi-res weather radar, tons of special interest forums, etc. But there is one item that destroys the interest for me. It costs big bucks to use the service. And I mean super big bucks. After my first months bill came to nearly forty bucks (that's on top of the five hours with the Starter Pack), I checked around and found that most people I talked to spent from $60-$100 a month or more. I don't know about you pardner, but that's outta my league. And please note: you don't get a bill...you just see an amount on your monthly credit card with no explanations. Want an an itemized bill? They'd be glad to send you one...for an extra $3.50. Remember those help functions? Well that's why they're there. They want you to spend all the time in the world figuring the stuff out.

SLO-POKES
OCT 84

(9)

Sometimes the help menus are more complex than the item you are trying to examine! I imagine that there are a group of people that sit around and figure out how to make the control functions as complex as possible. I can see it now..."Well Steve, what do you think about a ^RF to read messages here?" "OK, but let's disable the normal scroll commands and use some new ones...how about @T-9 at any ! prompt to continue?" The proverbial monkeys at typewriters would be proud.

The ATARI SIG boasts an impressive file area. There are a lot of downloadable programs here...games, terminal programs, utilities, etc. But as a new user, you'll quickly run into a problem: There is no way to download using IMODEM. You must get a copy of the terminal program called T-SCOPE. T-SCOPE allows you to use the COMPUSERVE transfer protocol. Catch-22 is that it's almost impossible to figure out how to download it and create a workable file. It comes in four binary sections which must be merged to create the working program. After many attempts, I gave up and got a copy from a friend. If you would like a copy, I'd be glad to spare you the grief and pass it on.

Perhaps my biggest gripe with the system besides its needless complexity is the "extra" charges. Like to try out the Shop-at-Home feature? There's an extra charge. Likewise for the Flight Plans, the software exchange, the Airlines Guide, most of the data bases, several of the games, all of the decent stock and bond features, and probably a lot more I haven't found yet. And we're talking some big money here...for instance, the Airlines Guide will set you back an additional $22.00 an hour during the off hours. Let's see, that would be $22.00 an hour for that, plus $5.00 an hour for the standard COMPUSERVE service, plus $2.00 an hour for TYMNET connection. That's a whopping $29.00 an hour to check on flights from LAX to NYC. And that's during the off hours. Want to check during normal business hours? How does $55.50 an hour sound? Travel Agents have nothing to fear here.

Nutshell: I like the service. I enjoy getting on the wires and looking around other BBS's and COMPUSERVE is one choice system. But I just don't have the bucks. Just a casual look around on the weekends is going to run me $20.00 to $30.00 a month and any serious usage of the games or other features would easily top the $100.00 mark. This system is like a house on the beach, a holiday to Tahiti, and a six-figure income; I'd like to have em, but I can't.

-T. Wiggins-
70416,377

AARI 9-84

(10)

>>>>> BASIC BASKET <<<<<

DIF FILES
by David Fuller

SYNFILE+ is the latest in filemanagement programs and in my opinion is the best. In the 3 years I have owned my ATARI, I have spent most of my time writing programs for data files. I am very familiar with ATARI's capabilities and limitations. SYNFILE+ makes use of all of ATARI's attributes, including expanded memory. I wrote a disk catalog program because the best filemanagement program was only able to search on the one main index field. The disk catalog program could search on three. I later modified the disk catalog program to be a VHS

HACK

FROM END OF THIS ARTICLE


```
920 DIM ANS$(255),INFIL$(17),FLDN$(240),LIN$(40),CHGFLD$(12),FNCT$(40),FLNAM$(17),OUTFIL$(17),OUT$(1)
930 DIM CHG$(255),INPT$(255),X$(1),STYPE$(2),FLD1$(255),FLD2$(255),FN(20),FT(20)
940 LIN$="------------------------------------":? LIN$:A=0:B=5:C=2
950 RETURN
```

D:CHECK DATA

```
10 DATA 230,128,122,788,388,327,310,273,285,637,402,901,642,802,425,6660
150 DATA 78,392,719,618,16,214,451,985,796,502,837,265,956,593,978,8400
300 DATA 530,921,318,985,985,984,458,842,415,902,619,502,952,583,149,10145
420 DATA 680,85,945,850,741,662,587,47,856,103,937,269,130,142,657,7691
500 DATA 376,677,183,266,56,594,859,259,733,314,18,908,790,252,299,6584
610 DATA 481,557,244,782,783,157,930,793,893,921,650,939,254,957,34,9375
690 DATA 908,187,131,370,836,799,872,307,836,363,613,493,798,21,836,8370
810 DATA 276,272,341,966,606,196,608,29,670,258,138,358,581,917,800,7016
940 DATA 914,612,1526
```

movie file in which 1250 records, the program, and the index, all resided on one disk. SYNFILE+, although the program is on a seperate disk, has the ability to search on sixteen fields and the size of your file is only limited by memory for the index. It can search across disks.,
Good-bye disk catalog, hello SYNFILE+. I decided I wanted my files in SYNFILE+. This meant I would first have to convert my files to DIF (Data Interchange FILE) files, so I wrote a program to do so.

DIF files were originally set up by the people who wrote Visicalc in order to have a universal file structure that could be used to transfer files between programs or between computers. Because of this structure (which I think was done a little backwards), it takes a long time to process data files. The DIF file is set up with rows and columns and the information is stored by column. All the information in the first column is written first, then the second column, and so on. When a data file is being converted, this means that the information is written by field. The data file has to be read from beginning to end once for each field. If you have 16 fields, it has to be read sixteen times. This is just to convert to DIF. Then the file has to be loaded into SYNFILE+. I did 72 records which took 45 minutes. The DIF file winds up being much larger than the source file. If you have a full Filemanager data disk, you will probably need 2 double density drives to convert it to a DIF file. Another way would be to create subfiles with Filemanager and do each subfile separately. In the program I have included a function to break down large files into two smaller files. For each field in your data file, there are at least 8 characters added to the DIF file plus some additional information to separate the fields. In addition, when SYNFILE+ converts the DIF file to it's own format, it has a minimum field length of 16 characters. This means if your data file has a field with 2 charaters, when SYNFILE+ converts it, it will contain 16 characters. This can be modified back to 2 characters with Modify Form and Merge functions. The only thing I found I didn't like about SYNFILE+ is the fact that it doesn't give you any idea what is going on when it is converting DIF files. It could take an hour, or it could take 6 hours. You never know until it is done.

This BASIC program will automatically convert filemanager files to DIF files and will also convert other DOS type files if you know the structure (number of fields,lengths,whether they are strings or numeric, and the number of records in the file). If you are converting a Filemanager file, simply insert the data disk and the program will give you a list of the files on the disk. You then select which file you want to convert. The program will show you a list of the field names to make sure you have selected the right files. You will then be asked how many records you want to copy. This allows you to create 2 smaller DIF files from a very large data file. If you enter the same number as the total records, the conversion will only create one file. If you enter a number that is less than the number of records in the file, the coverion will create one file with the number of records you entered and another file with the rest of the records. Next you will be asked for the name of the output file. After all these entries have been made, the program will proceed to write the DIF file, displaying the field and record numbers as it goes along.

At the beginning of the program you will be asked if you want to convert a Filemanager file or other. If you select other, you will be asked questions about the fields and file names. Then it will again start to create the DIF file.

Type in the program and save it to disk. Use Analog's D:CHECK program and the D:CHECK data after the program listing to verify that the program was typed in correctly. You might want to create or use a small Filemanager file first to make sure you understand how everything works before you try to convert a large file. This program will also be on the AARI BBS (401-521-4234).

Have fun

Dave

(12)

DIF CONVERTER PROGRAM

```
10 TRAP 880:GOSUB 900:? "K";TTL$
20 REM  GET FILE NAME
30 ? "          by David Fuller"
40 ? :? " How many Drives (1 or 2) ";:INPUT DRIVES
50 IF DRIVES(1 OR DRIVES)2 THEN GOTO 40
60 POSITION 2,6:? "1. Filemanager"
70 ? "2. Other"
80 ? :? "Enter Type of File ";:INPUT ANSR$
90 IF ANSR$()"1" AND ANSR$()"2" THEN GOTO 60
100 IF ANSR$="2" THEN TYP=2:GOTO 710
105 TYP=1
110 ? :? " Insert Filemanager Data Disk"
120 ? "   in Drive 1 and press RETURN ";:INPUT ANSR$
130 ? "K";TTL$:? :? "     Files on the Disk":? LIN$
140 INFIL$="D1:*.*":OPEN #1,6,0,INFIL$
150 INPUT #1;FLNAM$:IF FLNAM$(5,8)="FREE" THEN 180
160 IF FLNAM$(11,13)="DAT" THEN ? FLNAM$(1,8);"    ";FLNAM$(14,17)
170 GOTO 150
180 CLOSE #1:? LIN$:? " Which File to Convert ":? "     ";:INPUT ANSR$:INFIL$="D:":INFIL$(3)=ANSR$
190 INFIL$(LEN(INFIL$)+1)=".FMT"
200 REM  GET AND DISPLAY FIELDS
210 ? "K";TTL$:POKE 752,0
220 OPEN #1,4,0,INFIL$
230 INPUT #1;NUMFLD$
240 INPUT #1;FLD$
250 MAXLEN=0
260 FOR N=1 TO NUMFLD$:INPUT #1;A:FN(N)=A:IF A)MAXLEN THEN MAXLEN=A
270 NEXT N:A=0
280 CLOSE #1:LE=LEN(INFIL$):INFIL$(LE-3,LE)=".IDX":OPEN #1,4,0,INFIL$
290 INPUT #1;A:NUMR=A
300 FOR N=1 TO 4:INPUT #1;IDX1:NEXT N:INPUT #1;IDX2:INPUT #1;IDX3
310 ? :? "File ";CHR$(34);INFIL$(3,LEN(INFIL$)-4);CHR$(34);" has ";NUMR;" records"
320 ? :? ,"Field names"
330 ? LIN$:A=0:B=5:C=3
340 FOR N=1 TO NUMFLD$*12 STEP 12:A=A+1
350 B=B+1:IF A=11 THEN C=20:B=6
360 IF A=10 THEN C=2
370 POSITION C,B:? A;". ";FLD$(N,N+11)
380 NEXT N:POSITION 2,16:? LIN$
390 ? "  Is this the right File (Y/N) ";:INPUT ANSR$
400 IF ANSR$()"Y" THEN RUN
401 ? :? "How many records to copy ";:INPUT CPY
405 PASS=1:IF CPY(NUMR THEN PASS=2
406 START=1
410 INFIL$(LEN(INFIL$)-2)="DAT"
420 REM  WRITE DIF FILE
430 ? :? "  Enter name of output file":? "   up to 8 characters ";:INPUT ANSR$
440 OUTFIL$="D":OUTFIL$(2)=STR$(DRIVES):OUTFIL$(3)=":":OUTFIL$(4)=ANSR$:OUTFIL$(LEN(OUTFIL$)+1)=".DIF"
442 IF DRIVES=1 THEN GOTO 450
443 ? :? "Insert Desitnation disk "
445 ? "   in Drive 2 and press RETURN ";:INPUT ANSR$
450 ? "K";TTL$:? :? "     Processing..."
455 POSITION 5,8:? "Total Records: ";NUMR
460 POSITION 5,10:? "Field: ";:POSITION 5,12:? "Record #:";
470 OUT$=CHR$(34)
480 OPEN #2,8,0,OUTFIL$
481 IF PASS=2 AND START)1 THEN GOTO 486
485 CLOSE #1:OPEN #1,4,0,INFIL$:NOTE #1,SEC,BYTE
```

DIF CONVERTER AND MANY OTHER FINE PROGRAMS WRITTEN BY AARI MEMBERS ARE AVAILABLE ON DISK IN EXCHANGE OF PUBLIC DOMAIN SOFTWARE. IF INTERESTED CONTACT MAURICE LEBLANC

```
486 IF START()1 THEN CPY=NUMR
490 ? #2;"TABLE":? #2;"0,1":? #2;OUT$;OUT$:? #2;"VECTORS":? #2;"0,";STR$(CPY-START+1):? #2;OUT$;OUT$:? #2;"TUPLES"
500 ? #2;"0,";STR$(NUMFLD5):? #2;OUT$;OUT$
510 ? #2;"DATA":? #2;"0,0":? #2;OUT$;OUT$
511 A=0
515 IF TYP=2 THEN ST=0:FOR N=1 TO NUMFLD5:A=A+FN(N):NEXT N:GOTO 550
520 FOR N=1 TO NUMFLD5:FT(N)=0
530 IF FN(N))100 THEN FT(N)=1:FN(N)=14
532 A=A+FN(N)
540 NEXT N:ST=0
550 IF A)255 THEN ? "Total record length over 255 characters, can not convert ":END
555 FOR Z=1 TO NUMFLD5
560 IF TYP=2 THEN POSITION 13,10:? Z:GOTO 580
570 POSITION 13,10:? FLDN$(Z*12-11,Z*12);
580 ? #2;"-1,0":? #2;"DBT"
590 ST=ST+FN(Z)
600 FOR X=START TO CPY
610 POSITION 15,12:? X;"   ";
620 INPUT #1;INPT$
630 IF FT(Z)=0 THEN ? #2;"-1,0":? #2;OUT$;INPT$(ST-FN(Z)+1,ST);OUT$
640 IF FT(Z)=1 THEN ? #2;"0,";VAL(INPT$(ST-FN(Z)+1,ST)):? #2;"V"
650 NEXT X
655 IF Z=NUMFLD5 THEN NOTE #1,SEC,BYTE
660 POINT #1,SEC,BYTE
670 NEXT Z
680 ? #2;"-1,0":? #2;"E00":CLOSE #2
681 IF CPY=NUMR THEN GOTO 690
682 ? :? :? "Do you want to copy the rest of the":? "  records ";:INPUT ANSR$:IF ANSR$(1,1)()"Y" THEN 690
683 POINT #1,SEC,BYTE
684 START=CPY+1
685 IF DRIVES=2 THEN ? :? "Insert new destination disk ":? " and press RETURN  ";:INPUT ANSR$
686 ? "A";TTL$:GOTO 430
690 POSITION 5,19:? " DONE ":END
700 REM SET UP NEW FILE INFO
710 ? "A";TTL$:? "   Convert Other File"
720 ? :? "How many Fields in File ";:INPUT NUMFLD5
725 ? :? "How many Records in File ";:INPUT NUMR:?
726 ? :? "How many records to copy ";:INPUT CPY:?
728 IF CPY(NUMR THEN PASS=2
730 FOR Z=1 TO NUMFLD5
740 ? "Length Field #";Z;:INPUT X:FN(Z)=X
750 ? "Type of Field (S)tring/(N)umber) ";:INPUT ANSR$:IF ANSR$(1,1)()"S" AND ANSR$(1,1)()"N" THEN ? "S OR N":GOTO 750
760 IF ANSR$(1,1)="N" THEN FT(Z)=1:GOTO 780
770 FT(Z)=0
780 NEXT Z
790 ? "A";TTL$:? :? "  #   Length  Type":? LIN$
800 FOR Z=1 TO NUMFLD5:? "  ";Z;"   ";FN(Z);"      ";
810 IF FT(Z)=0 THEN ? "String"
820 IF FT(Z)=1 THEN ? "Number"
830 NEXT Z:? LIN$
840 ? :? "Are all entries Correct (Y/N) ";:INPUT ANSR$:IF ANSR$(1,1)()"Y" THEN GOTO 710
850 ? "A";TTL$:? :? "Enter name of Source File":? " example: TEST.DAT   ";:INPUT ANSR$
860 INFIL$="D:":INFIL$(3)=ANSR$
865 START=1
870 ? "A";TTL$:GOTO 430
880 ERR=PEEK(195):ERLIN=PEEK(186)+256*PEEK(187):? :? "          ERROR ";ERR;" AT LINE ";ERLIN
890 ? :? "  Press RETURN to continue ";:INPUT ANSR$
895 RUN
900 REM SET UP VARIABLES
910 ? "A";:DIM TTL$(34):TTL$=" CONVERT TO I:I FILE UTILITY "
```

(13)

ENTERING MACHINE PROGRAMS INTO BASIC STRINGS

ABACUS SEPT 84 (14)

by
I.F. Carlstrom
Cleveland Hts., Ohio, 44106

ATARI Basic allows you to define very long strings, which are an ideal way to include relocatable machine routines in a Basic program. All symbols except the double quotes and the RETURN (ASCII code 155) can be included in a string defined inside quotation marks. If care is taken in writing an assembly program, its machine code can be defined by such a string of symbols inside quotation marks and then loaded virtually instantaneously when the Basic program is run. A good example of this technique can be found in my modifications of the word processing program, SCRIPTOR.

The following Basic program can take a machine language object code file, such as would be created by the SAVE function from the ASSEMBLER-EDITOR cartridge, and create a disk file with lines of a Basic program defining the corresponding string. This file can then be directly ENTERed into your program. All relevant details such as line numbers, name of the string, etc., can be controlled, so you will find this to be a quick and easy way to enter or change such files. Other files, such as redefined character sets from font generators could also be entered in this way.

After running the program, you will first be asked for the file name of the machine language object program (unless you specify otherwise, the extender is assumed to be ".OBJ"). The object program will then be entered and scanned for any unacceptable symbols. Watch the program to see if any such undesirable symbols are listed. If not, you can store your program in a string without taking special steps. After the program has been entered and checked, you will be asked to Press a Key to give you time to check for bad symbols. The string defining the machine program will then be printed out on the screen and you will be asked if you wish to eliminate the first six bytes. Press "Y" if this is a typical file, such as one created by an assembler, which has an initial running address in these bytes. You will next have an opportunity to select variable names, starting line numbers, etc. (You can just press RETURN if the pre-selected names are acceptable.) Finally, a disk file will be created with lines of a Basic program that defines a string that holds your machine program. You now need only ENTER this file into your program. An exact duplicate of the disk file will be printed out on the screen so that you will know what lines are being defined in the file.

Note that the string defining the program will automatically be DIMensioned to the correct value, but if more than one line is required to store the program, an additional string (such as "AD$") is used. This string should be DIMensioned to an length of 102 somewhere previously in your Basic program. (This additional string variable can be used in entering several machine routines into your Basic program without DIMensioning it more than one time.)

When entering the program note that " ***** " is " ##### " in inverse video and the arrows such as " ↑ " are obtained by first typing ESC.

```
80 REM *****  ASM ENTRY PROGRAM *****
82 REM *****  ALLOWS ENTRY OF OBJECT LANGUAGE PROGRAMS INTO BASIC STRINGS
84 REM
86 REM
94 REM *****  COPYRIGHT 1984, BY I.F. CARLSTROM
95 REM *****  CLEVELAND HTS., OHIO,44106
96 REM *****  THE FOLLOWING PROGRAM MAY BE FREELY COPIED FOR PERSONAL USE, BUT NOT FOR COMMERCIAL PURPOSES.
98 REM
99 REM
0100 GRAPHICS 0:? :? "          TEST PROGRAM":? "  for programs up to 2000 bytes":?
0104 DETRAP=44444:DIM DES$(10),B$(20),C$(20),A$(2000),AD$(20),PR$(20)
0106 AD$="AD$"
0110 POSITION 2,7:? " (AUTOMATIC EXTENSION-  .OBJ)***"
0120 PRINT "ENTER PROGRAM FILE NAME";:INPUT B$:C$="D:":C$(LEN(C$)+1)=B$:POSITION 2,9
0130 TRAP 135:C$(LEN(C$)+1)=".OBJ":CLOSE #3:OPEN #3,4,0,C$:GOTO 140
0135 PRINT C$;" CAN'T BE LOADED              ":GOTO 110
0140 TRAP 155:I=1:? "LIST OF UNACCEPTABLE VALUES":? "POS.","NUMBER"
0145 GET #3,A:IF (A=34 OR A=155) THEN PRINT I,A
0150 A$(I,I)=CHR$(A):I=I+1:GOTO 145
0155 TRAP DETRAP:? :? "PRESS RETURN TO CONTINUE.":INPUT DES$
```

(15)

```
0160 IF A6(LEN(A6))="" THEN A6=A6(1,LEN(A6)-1):GOTO 160
0165 L=LEN(A6):K=INT(L/100)+1:IF 100*K=L THEN K=K-1
0170 GRAPHICS 0:? :? :? " THE PROGRAM ";C$
0180 ? " IS OF LENGTH ";L;".":? "THE FOLLOWING IS A LISTING:"
0300 FOR I=1 TO L:PRINT CHR$(27);A6(I,I);:NEXT I:? :?
0310 ? :? :? "ELIMINATE SIX BYTE LEADER (Y/N)";:INPUT DES$:IF DES$="Y" THEN A6=A6(7)
0320 GRAPHICS 0:? :? :C$(LEN(C$)-2)="ENT":? "SAVE FILE AS ";C$:? "**************";:INPUT C$
0330 TRAP 340:CLOSE #3:OPEN #3,4,0,C$
0335 TRAP DETRAP:? "REPLACE ";C$;:INPUT DES$:IF DES$<>"Y" THEN GOTO 320
0340 CLOSE #3:OPEN #3,8,0,C$
0350 ? :? "USE ADDING STRING AS ";AD$
0360 ? "**********************";:INPUT AD$
0370 PR$=C$(3,LEN(C$)-4):PR$(LEN(PR$)+1)="$"
0380 ? :? "USE MACHINE STRING AS ";PR$
0390 ? "**********************";:INPUT PR$
0400 TRAP 400:? :? "ENTER STARTING NUMBER FOR PROGRAM":? "LISTING";:INPUT SN
0410 TRAP 410:? "ENTER INCREMENT NUMBER FOR PROGRAM":? "LISTING";:INPUT IN
0420 TRAP DETRAP
0500 ? :? SN;" DIM ";PR$;"(";L;")"
0505 ? #3;SN;" DIM ";PR$;"(";L;")"
0510 IF K=1 THEN GOTO 210
0520 PRINT SN+IN;" ";PR$;"=";CHR$(34);
0525 PRINT #3;PRINT #3;SN+IN;" ";PR$;"=";CHR$(34);
0530 FOR J=1 TO 100:PRINT CHR$(27);A6(J,J);:NEXT J:?
0535 FOR J=1 TO 100:PRINT #3;A6(J,J);:NEXT J:?
0540 IF K=2 THEN 580
0550 FOR I=1 TO K-2:PRINT SN+2*IN*(I);" ";AD$;"=";CHR$(34);
0555 FOR I=1 TO K-2:PRINT #3:PRINT #3;SN+2*IN*(I);" ";AD$;"=";CHR$(34);
0560 FOR J=100*I+1 TO 100*I+100:PRINT CHR$(27);A6(J,J);:NEXT J:?
0565 FOR J=100*I+1 TO 100*I+100:PRINT #3;A6(J,J);:NEXT J:?
0570 PRINT SN+2*IN*I+IN;" ";PR$;"(LEN(";PR$;")+1)="";AD$
0575 PRINT #3:PRINT #3;SN+2*IN*I+IN;" ";PR$;"(LEN(";PR$;")+1)="";AD$:NEXT I
0580 I=K-1:PRINT SN+2*IN*I;" ";AD$;"=";CHR$(34);
0585 PRINT #3:PRINT #3;SN+2*IN*I;" ";AD$;"=";CHR$(34);
0590 FOR J=100*I+1 TO LEN(A6):PRINT CHR$(27);A6(J,J);:NEXT J:?
0595 FOR J=100*I+1 TO LEN(A6):PRINT #3;A6(J,J);:NEXT J:?
0600 PRINT SN+2*IN*I+IN;" ";PR$;"(LEN(";PR$;")+1)="";AD$
0605 PRINT #3:PRINT #3;SN+2*IN*I+IN;" ";PR$;"(LEN(";PR$;")+1)="";AD$
```

computer stars

There is a current trend to hire certain celebrities to identify with certain computers and act as spokespersons for their particular computer in television commercials and newspaper ads. The present line-up and the reasons for hiring the celebrity are as follows:

Alan Alda ATARI
He has high credibility and that sincerity will be communicated to consumers. Our aim is to convey the image of family entertainment and education through technology.

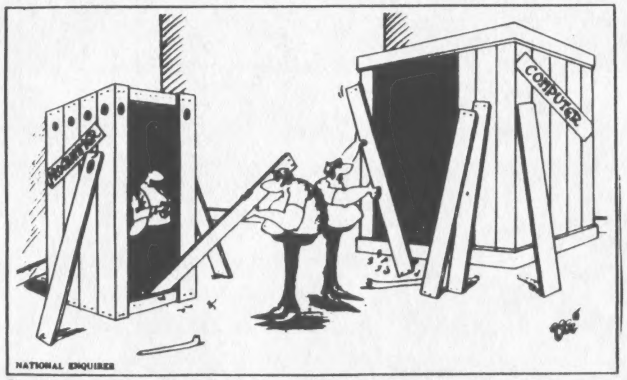
Roger Moore Spectravideo
He's played James Bond and universally represents high technology.

Bill Cosby Texas Instrument
He's got a PH.D. in education and also he's got good rapport with kids and adults.

Dick Cavett Apple
He represents the "everyman" and conveys an image of being approchable, friendly, and innovative.

Isaac Asimov Radio Shack
He's so well known as a futurist, and we want to be seen as a company with products of the future.

William Shatner Commodore
Because he is identified with Star Trek and seen as a commander who makes good decisions in the area of high technology.



(16)

GRAPHICS TABLET DRAWING TIPS

By Marion Delahan

This month I've been asked to give some tips on drawing with the various graphics tablets that are on the market. Specifically I'll be dealing with the various incarnations of the Micro-Illustrator type software that comes with them.

To find the center of the screen use the four-way mirror feature, the point cursor, and the frame drawing utility. It is best to use one of the three colors mixed with the background color as this gives a nice dotted line that will not easily be confused with an intentional line. The guidelines can then be erased when they are no longer needed. Put the stylus in one corner of the screen as far up and over as the tablet will register. Pull it slowly toward the center without lifting it. When the four lines just meet in the center, freeze the display by pushing the button. On the Atari tablet this will give a thicker line as the axes. This is because there is an even number of points vertically and horizontally. There are two lines in the center in both directions. Now that the center of the screen is clear it is a simple task to draw any symmetrical object by switching the mirror function as needed.

The files as stored on disk are in a condensed format that is not compatible with other programs. The data can, however, be translated into a standard file in several ways. One way is to use the insert key. With the picture on the screen, press the insert key. A file will be created on disk called "Picture". Be sure to rename the file if you intend to make another conversion on the same disk. If you do not have the graphics tablet handy you may run the KOALA1.BAS program from last month's KEEPING PACE. The program also works with the Atari tablet, although it runs quite slowly. You may even think it isn't working; it is, just very slowly. When we found the program, it took about six minutes from start to finish. Turning off the screen during the file conversion produced a program that ran in about four and a half minutes. With BASICXL it took just under two. Anyone who is interested in speeding it up a little more will be able to cut the time further by turning off the write verify on the disk with a POKE 1913,80. To turn the verify back on POKE 1913,87.

Now you have a 62 sector file on your disk. What do you do with it? It can now be loaded with a graphics loader. Koala provided one. There are others available, including one that comes with PRINTWIZ (from Allen Macroware). You could also use Micropainter to touch the picture up. The August '83 issue of ANTIC contains a program called PICUTE that allows you to move a picture around on the screen, to add text or to add borders. This also allows you to share disk pictures with others who don't have the tablet. Another thing that you can do with a picture is print it. For most of us this means converting a beautiful color masterpiece into a hopelessly black and white picture but in some applications that may not matter and besides it is the best we can do. Printwiz does a good job of dumping the converted file. I recently picked up some information on Compuserve that there is a new version of PRINTWIZ which will dump an unconverted file. It is available from the publisher as an update for around $10.

A printed picture might be worth a thousand words, but a few words of caution are in order, if the final print is going to appear as the budding Picasso wishes it to. Most print dump programs use some form of control over the shading of the picture by assigning different densities of print to the different color registers. It doesn't matter what the color of the original picture was only what color register has been used to create a given area. In PRINTWIZ, for instance, whatever color is in register 0 will be blank, and whatever color is in register 3 will be black. Before you print your picture and get a surprise, make sure you know what your dump program is doing with the registers.

The new graphics tablets are really remarkable tools and many things that were difficult to do before can be fairly easily accomplished with them. It does take a bit of persistence, though, to be able to use the new tools with facility. Practice, persevere and you too can turn out artwork that will at least satisfy your own artistic bent.

TCC
11-84 (17)

INTERNAL KEYCODES

KEYBOARD

A 63	B 21	C 18	D 58	E 42	F 56	G 61
H 57	I 13	J 1	K 5	L 0	M 37	N 35
O 8	P 10	Q 47	R 40	S 62	T 45	U 11
V 16	W 46	X 22	Y 43	Z 23	1 31	2 30
3 26	4 24	5 29	6 27	7 51	8 53	9 48
0 50	- 14	= 15	; 2	+ 6	* 7	/ 38
. 34	, 32	< 54	>55			
BS 52	RET. 12	CAPS 60	LOGO 39	ESC 28	TAB 44	SPACE 33

CONTROL KEYBOARD

A 191	B 149	C 146	D 186	E 170	F 184	G 189
H 185	I 141	J129	K 133	L 128	M 165	N 163
O 136	P 138	Q 175	R 168	S 190	T 173	U 139
V 144	W 174	X 150	Y 171	Z 151	1 ---	2 158
3 154	4 152	5 157	6 155	7 179	8 181	9 176
0 178	- 142	= 143	; 130	+ 134	* 135	/ 166
. 162	, 160	< 182	> 183			
BS 180	RET. 140	CAPS 188	LOGO 167	ESC 156	TAB 172	SPACE 161

SHIFT KEYBOARD

A 127	B 85	C 82	D 122	E 106	F 120	G 125
H 121	I 77	J 65	K 69	L 64	M 101	N 99
O 72	P 74	Q 111	R 104	S 126	T 109	U 75
V 80	W 110	X 86	Y 107	Z 87	1 95	2 94
3 90	4 88	5 93	6 91	7 115	8 117	9 112
0 114	- 78	= 79	; 66	+ 70	* 71	/ 102
. 98	, 96	< 118	> 119			
BS 116	RET. 76	CAPS 124	LOGO 103	ESC 92	TAB 108	SPACE 97

CONTROL SHIFT KEYBOARD

A 255	B ---	C ---	D 250	E 234	F 248	G 253
H 249	I 205	J ---	K ---	L ---	M 229	N 227
O 200	P 202	Q 239	R 232	S 254	T 237	U 203
V ---	W 238	X ---	Y 235	Z ---	1 223	2 222
3 218	4 216	5 221	6 219	7 243	8 245	9 240
0 178	- 206	= 207	; ---	+ ---	* ---	/ 230
. 226	, 224	< 246	> 247			
BS 244	RET. 204	CAPS 252	LOGO 231	ESC 220	TAB 236	SPACE 225

(18)

ATARI INTERRUPT STRUCTURE and THE SERIAL PORT

by Donald B. Wilcox

Interrupts permit events to gain immediate access to the computer without the necessity of your software constantly checking (polling) to determine whether or not a device needs to communicate with the computer. When an interrupt is enabled (able to respond), it will automatically gain access to the computer when necessary, perform its task, then return control to the normal program that was running before the interrupt. As an example, if someone calls you (interrupts you) on the telephone, the bell rings for your attention to let you know that an interrupt has occurred. You are not required to constantly pick up the phone to see if someone wants to talk to you. Each type of interrupt has its own location in memory where its program is stored. A pointer (vector) is a special memory location that contains the address of the program to run. When the interrupt occurs, the computer checks the address (vector) associated with that particular interrupt, then transfers control to the location indicated by the vector pointer. After the interrupt program is finished with its task, it returns control to the program that was running before the interrupt, much the same as you would return to your prior activity after answering a telephone call.

This article is oriented primarily to the atari serial port which is a thirteen pin connector for communicating with peripheral devices. This is the port to which you normally connect your disk drive, cassette recorder or any other serial transmission device. Figure 1. depicts the pin configuration of the serial port.

2 4 6 8 10 12

1 3 5 7 9 11 13

Figure 1: Serial Port

This article discusses the use of pins 9 and 13 which are currently unused by any standard devices and which are essentially ignored by the operating system. Both of these pins are part of the interrupt processing structure. We will digress temporarily to provide a cursory overview of the Atari interrupt process to enhance our understanding before discussing the Serial Input, Output (SIO) port. There are two types of interrupts available at the processor level. The first is a nonmaskable interrupt (NMI), the second is a maskable (IRQ) interrupt.

The Nonmaskable interrupts are handled by the operating system. These include the SYSTEM RESET, DISPLAY LIST, and VERTICAL BLANK interrupts. Although there are no vectors available for the SYSTEM RESET, it can be trapped by using the DOSINI at locations 12, 13. In a disk drive environment, DOS is initialized whenever SYSTEM RESET is activated, thus the DOS initialization vector can be used to trap SYSTEM RESET. (See De Re Atari, Chapter 8, Memory Management subsection for additional information.) The DISPLAY LIST is not used by the operating system, but it is vectored for control through the ANTIC chip. If the Display List Interrupt (DLI) is enabled by the user, then you can have the vector at locations at 512, 513 point to your DLI routine. The Vertical Blank Interrupt (VBI) is vectored at two locations, one is for the immediate mode interrupt and the second is for the deferred mode interrupt. Each of these can be intercepted by the user to activate a small user written module. The immediate mode is vectored at locations 546, 547; the deferred mode is vectored at locations 548, 549. Although these NMI's cannot be masked at the 6502 chip level, the Atari ANTIC chip can be used to enable and disable both the DLI and the VBI. (See De Re Atari, Chapter 8, 'The NMI Handler' subsection.)

The maskable interrupts (IRQ) are handled by the Peripheral Interface Adapter (PIA) chip and the Atari POKEY chip. Each of the IRQ interrupts are vectored and are accessible to the user. (The BREAK key was not vectored in the old Operating System, version A, but is vectored in revision B).

A second digression may be in order at this point for those who are unfamiliar with the concepts of vectors and interrupt processing. Each interrupt module is activated either by the operating system (automatically) or by the user. The user can activate an interrupt in their program software or by a manual input

(Continued on page 6)

ATARI INTERRUPT STRUCTURE and THE SERIAL PORT (Continued)

from the keyboard, joystick ports or SIO port. As an example, each time you press a key on the keyboard, you activate the keyboard interrupt which is vectored at locations 520, 521. This means that the computer looks at the memory address stored in locations 520, 521 and then transfers control to that part of the memory. You could write a routine, store it at a specific memory location, put the address of your routine at locations 520, 521; then when a key is pressed on the keyboard, the computer will transfer directly to your program. This technique can be used to disable the break key with operating system version B. The Break key is vectored at 566, 567. Normally, these locations point to (are vectored to) address 59220 which is the beginning of the normal break key routine. If you change the values at locations 566, 567 to point to 59279, the break key will be inoperable. The reason is that location 59279 contains the assembly language mnemonics: PLA, RTI. This causes the computer to do nothing but return to where is was before the interrupt occurred. If desired you could change the break key vector to point to a special software routine to be activated when the break key is pressed. (Note: The console keys; OPTION, SELECT & START are not available though the interrupt process, but the user can write software that monitors their status).

Finally, we come back to pins 9 and 13 of the SIO port. Both of these pins are vectored and are interrupt activated. The activation of the interrupt process is caused by (triggered by) a falling edge (voltage suddenly goes low from 5 volts to 0 volts). This falling edge trigger is normally supplied or caused by an attached peripheral device connected to the SIO port. The attached device would have to be designed to provide this falling edge as well as conform to the other transmission parameters associated with the transfer of serial data between the Atari and attached devices. You can create your own interrupt manually by connecting a wire to either pin 9 or 13 and touching it to another wire connected to a GROUND pin (pins 4 & 6 are GROUND pins). You will have the problem of 'bounce', that is, a manual attempt to touch two wires together creates in reality, dozens or even hundreds of touches, creating an interrupt each touch. Normally, falling edges are generated electronically to produce the equivalent of just one touch of the wires. You can minimize this problem by including in your interrupt driven software, a long delay before reactivating the interrupt vectors.

To enable or disable (activate or deactivate) interrupt pins 9 and 13 at the SIO port, you must use respectively Port A Control (PACTL at location 54018) for pin 9 and use Port B Control (PBCTL at location 54019). For those of you unfamiliar with bits and bytes, a short explanation may be helpful. A byte is a single character or a number such as a '1' or a '3'. Each key on the keyboard can generate a byte of data including all the special characters. Each byte is comprised of 8 bits. A bit can be either a '1' or a '0', only these two values (binary base) are recognized by the computer. All programs and data are represented by a string of '1's and '0's. The computer can correlate different patterns of '1's and '0's with different characters. There are 8 bits in each byte. Since each bit can be either a '1' or a '0', there are 2*2*2*2*2*2*2*2 or 65536 different patterns of 8 bits possible. 1 K (kilobyte) equals 2*2*2*2*2*2*2*2*2*2 (1024) bytes. 65536/1024 = 64 K or the maximum number of different 8 bit patterns that can be recognized, therefore the maximum number of memory locations that can be addressed in an 8 bit machine such as the 6502 which is the microprocessor chip found in the Atari, Apple and Commodore. Figure 2 illustrates the naming of the bits in each byte.

7 6 5 4 3 2 1 0

Figure 2. Bit Names

Returning now to our discussion of interrupts, pin 9 is enabled by making bit 0 (lowest order bit) equal to '1'. PACTL normally contains the value 60. You can enable pin 9 by poking a 61 into location 54018. Analogously, you can enable pin 13 by poking a 61 into location 54019. It may seem strange to use the locations that control the joystick parallel ports to also enable the serial port pins 9 & 13, but this is because the PIA chip was engineered by Atari in this

(Continued on page 8)

THE INTERRUPT STRUCTURE and THE SERIAL PORT (Continued)

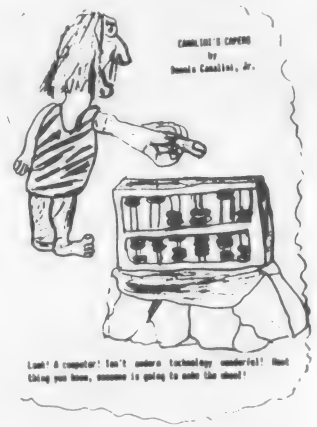
manner. (Maybe it is because there were no more bits available at the normal IRQ enable location 53774 which is used to control all the other maskable (IRQ) interrupts).

In reality, there is another mode of operation for pins 9 and 13 that does not require enabling the vectored interrupts and is readily available from software written in BASIC. For those of you who have some competency in ASSEMBLY language, the above should be sufficient to point you in the right direction for utilizing these special interrupt driven pins. (For more details See Mapping The Atari by Ian Chadwick, Atari Hardware Manual, Operating System Manual and Operating System Source Code and De Re Atari). For those of you who have yet to venture into the esoteric world of ASSEMBLY language, I offer the following technique for using pins 9 & 13 from BASIC.

Even when pins 9 & 13 have not been enabled by setting bit 0 to a '1' to respond to vectored interrupts, they do nevertheless take notice that an interrupt occurred. When an interrupt occurs at pin 9, bit 7 (highest order bit) of location 54018 (PACTL) is set to a '1' even though the interrupt vectors are not enabled. Bit 7 is called the STATUS bit. This status bit will remain set to '1' until the PACTL register (memory location) is read. PACTL usually contains the value 60, bits 3,4,5, & 6 are each set to '1'. If the enable bit (bit '0') remains '0' and an interrupt occurs, then bit 7 is set to '1'. This creates a value of 188 (original value of 60 plus 128 from bit 7). Reading (peeking) this register (PACTL) will set bit 7 back to zero. The same process applies to pin 13 using however location 54019 (PBCTL) instead.

The following BASIC program demonstrates this process.

```
100 REM PIN 13 INTERRUPT DEMO FOR MILATARI
NEWSLETTER
110 REM D.B.WILCOX 9-6-84
120 REM CONNECT ONE WIRE TO PIN 13 OF THE SIO PORT
130 REM (I/O CONNECTORS ON BACK OF DISK DRIVE)
140 REM IF NO DISK DRIVE, THEN CONNECT TO SIO PORT
ON COMPUTER
150 REM WHERE YOU NORMALLY CONNECT THE DISK DRIVE.
160 REM CONNECT OTHER WIRE TO PIN 6 (GROUND)
170 REM BASIC INTERRUPT PROGRAM BEGINS AT LINE 200
200 PRINT CHR$(125);"TOUCH WIRES TOGETHER TO
INTERRUPT"
210 PRINT PEEK(54019):REM PORT B CONTROL REGISTER
220 IF PEEK(54019)<>188 THEN 220:REM WAIT HERE FOR
INTERRUPT
230 IF PEEK(54019)=188 THEN PRINT
CHR$(125);CHR$(253);CHR$(253)
240 PRINT "INTERRUPT OCCURRED"
250 PRINT PEEK(54019):GOTO 200
```

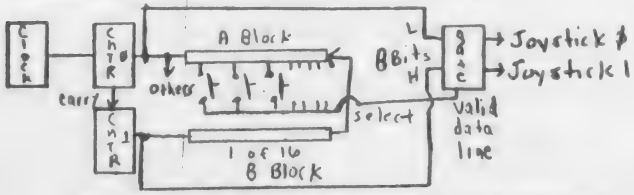


Look! A computer! Isn't modern technology wonderful! Next thing you know, someone is going to make the wheel!

AACE SEP 84

the Hobby Shop
by Rick Detlefsen

Last time I talked about using the joystick ports for reading the switches of the joystick and keypad. I have had inquiries as to how to do it so this month I will present a block diagram on this. Next month I will present the actual circuit to do it.



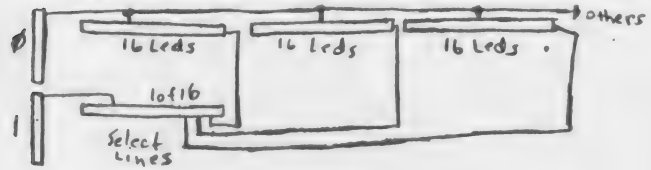
Basically you set up a clock circuit to scan each switch. Each switch is an 'A' circuit. Each sixteen switches is a 'B' block. Each 256 switches is a 'C' block. The result of the scanning is placed in the joystick port if a valid switch is pressed. Switch number zero is not used.

As the counter increments, a voltage is applied to each switch. If a switch is pressed, the circuit then assumes that the current clock number is to be read by the port. In the future, I will enhance this by using switch 255 for a special purpose.

Now then, to use the ports for output, just follow these steps:A(B)- POKE 54018,56(54019,56)-I/O control, POKE 54016,255(54017,255)-set up for output, POKE 54018,60(54019,60)-the following is data, POKE 54016,DATA(54017,DATA). To change to input-follow the above only instead of POKE 54016,255(54107) use POKE 54016,0(54017,0).

The block diagram below shows the procedure. Each four bits of the port is used to drive a 1-of-16 decoder. Only one led can be lit at a time, althought high speed may cause more than one to look lit.

Next month, I will start to present the actual circuits. BYE.



[B]

NULL MODEM

Knarf's Korner

By P.A.C.E.'s West Coast Correspondent,
Frank Nagle

Hello again from Atariland! A recent note on SIG*Atari related one persons use of the Atari as a backup system for data storage from a Radio Shack TRS-100. Having had a NEC PC8201A for a few months, I also had used the Atari for the same purpose - High Speed Mass Storage. Using the NEC is a lot simpler than the TRS-100. The NEC has an RS232 - 25 pin connector on the back as standard equipment. By taking the 25 pin connector currently in use between the Atari 850 interface and my modem, and installing a Double-Pole-Double-Throw (DPDT) switch on lines 2 and 3, I effectively created what is known as a "NULL" Modem between the Atari and the NEC. I have been able to transfer data to the Atari at 9600 baud!!!!, but the reverse, downloading to the NEC, is restricted to 1200 baud because of storage time necessary on the NEC. I use either ANODEM or CHAMELEON on the Atari, and the built in telecommunications program on the NEC to perform the system to system transfer. The NEC uses ASCII with no control characters, so the transfer is very easy, and requires no intervention or corrections to the data.

To use this simple device install the wiring changes as follows:

WIRING DIAGRAM

(NULL MODEM CABLE)

```
      [----------¦ ¦---]
 9 pin[----------¦ ¦---] 25 pin
to 850[----------¦ ¦---] to modem
                  ¦
            cable cut here
```

1 - Identify wires used on pins 2 and 3 of the 25 pin connector

2 - Cut cable open (be careful!! only the outside cover)

3 - Identify the 2 wires used in step one (1)

4 - Cut the two wires.

5 - Wire in the DPDT switch

```
pin 2 -----a      a------pin 2

pin 3 -----c      d------pin 3


          switch

        u*  v*  w*
  Left            Right
        x*  y*  z*
```

6 - connect a to v*
7 - connect b to w* and x*
8 - connect c to y*
9 - connect d to z* and u*

Pushing the switch to the right gives you normal operation with the modem, while pushing the switch to the left gives you a "NULL" Modem operation.

Pin 2 is normally = to send and Pin 3 is normally = to receive. Normal operation over telephone lines reverses 2 and 3 between the two locations using modems. When this step is removed it is necessary to reverse the lines locally for two computers to communicate.

Once the change is made, it is just a matter of removing the cable from the modem, pushing the switch to reverse lines 2 and 3, and connecting the cable to the NEC (or any similar type device). To connect the Atari back to the MODEM, merely push the switch the other way and the job is complete. Have fun with your new micro to mainframe connection.

Happy Atarling!!!

P.S. As I write this, I'm currently modifying the AMIS BBS system so I can run it locally and not have to operate on two keyboards. I will let you know next month how I make out.

>>>>> BILL'S KITCHEN <<<<<

BY BILL DWYER

PARTS FOR ATARI

GOOD NEWS

I JUST HEARD FROM BRAD KODA OF BEST ELECTRONICS IN CAL. YOU CAN GET FROM HIM A BRAND NEW 48K 800 COMPUTER FOR $180 OR AN 810 DRIVE FOR $265. NEED A FULLT STUFFED MOTHER BOARD FOR YOUR 800? HE'S GOT THEM FOR $24 OR ONE WITH JUST THE POKEY CHIP ON IT FOR $15. GOT AN OLD 800 AN NEED A GTIA CHIP THAT WILL COST YOU $8.

SOME OTHER ITEMS ARE:

1771 DISK CONTROLLER CHIP	9
16K RAM CARDS COMPLETE	15
BLANK ATARI 8+16 GAME BOARDS	2+4
EMPTY ATARI GAME CARTS	5
GENERIC DISKS SS SD 10PK	12
" " DS DD "	15
ATARI MODEM CABLES(THIS IS WHAT I USE ON MY HAYES FOR AMODEM OR THE BBS)	12
ATARI I/O CABLES	12
MICROSOFT CART	25
ASSEMBLER EDITOR CART	20
ATARI KEYPAD	90

SEVERAL OF US MET BRAD AT THE COMPUTER SHOW IN WOBURN MASS. A FEW MONTHS AGO. WE WERE VERY IMPRESSED WITH HIS KNOWLEDGE AND HIS PRICES AND THE FACT THAT BASICALLY HE IS A REAL NICE GUY TO DEAL WITH. THIS IS NOT WRITTEN AS AN ADVERTISEMENT FOR HIM BUT AS A SERVICE TO ATARI OWNERS WHO HAVE A TOUGH TIME FINDING REPLACEMENT PARTS AND CERTAIN HARD TO GET PEICES OF EQUIPMENT AT GOOD PRICES.

I HAVE SEVERAL OF HIS PRICE SHEETS IF YOU WANT ONE SEE ME OR GET IN TOUCH WITH HIM. THE LIST IS CONSTANTLY IN FLUX DUE TO AVAILABILITY.

CALL OR WRITE:

BEST ELECTRONICS
2021 THE ALMEDIA SUITE 290
SAN JOSE,CAL.95126
408-243-6950

TELL HIM (BRAD KODA) HOW YOU HEARD OF HIM.HE WILL BE AT THE SHOW AGAIN ON SEPT 28-29 .

NEXT THING COOKING IS A SIMPLE SHORT PROGRAM THAT I PUT TOGETHER TO HELP MY BROTHER'S CHILDREN LEARN THE MULTIPLICATION AND ADDITION TABLES.IT IS NOT VERY POLISHED SINCE IT DOES VERY LITTLE ERROR CHECKING BECAUSE IT IS JUST FOR HOME USE. IF IT GETS AN INPUT ERROR IT DIES BUT ALL YOU HAVE TO DO IS TELL IT TO RUN AND YOU'RE BACK IN BUSINESS. IF YOU GIVE A WRONG ANSWER IT TELLS YOU WHAT THE RIGHT ANSWER SHOULD BE AND THEN REPEATS THE QUESTION TO MAKE SURE YOU HAVE IT.HERE IS THE LISTING:

---------- HACK ----------

ATARI HARDWARE UPDATE

PARTS FOR ATARI

Here is some good news in reference to ATARI. Owners of 600XL and 800XL can get the new version of BASIC Revision C Free from Atari. Just send your Name, Serial Number, and a copy of the Sales slip to:

ATARI CORPORATION
390 Caribbean
Sunnyvale, CA 94089

For owners of the 400 and 800 the Revision C cartridge may be obtained by sending a check for $15.00 to the same address.()

The following is reprinted from Current Notes Oct. 1984, with our thanks. (ED.)

Spare Parts & Kits! That's right! If you need spare parts for your 810 drive or would like to build an ATARI 400 or 800 from a kit then you should contact: CENTURIAN ENTERPRISES, P.O. Box 3233, San Luis Obispo, CA. 93403 (805) 544-6616 (Cash, Check, MO or COD - no cards!)

All merchandise is warranteed for at least 180 days & they guarantee your satisfaction on any purchased item. Quantity discounts are available on most products at 6 and 11 pieces. Typical prices for the 400 kit are $50 with 16k RAM and power supply (does not include the plastic case). The 800 with 16K is $95 and does NOT include the case or keyboard. Mother boards, ROM boards, etc are also available separately for about $25 to $50. Disk drive mechanisms from $100 and a complete 810 kit with power supply is available for $250. Assembled: $285.

Cartridges for BASIC, the Assembler, and games range from $10 to $20 with and without the plastic shell. Also, a variety of peripherals from various manufacturers are available. Believe it or not, many custom ATARI chips and special IC are also in stock! Service manuals are available from: ELECTRONIC DIMENSIONS, P.O. Box 56, Auburn, CA 95603 (916) 637-4634

****

PROGRAM DOCUMENTATION

by Rolly Herman

I have discovered a fairly simple method of including written text for instructions to appear on the screen of your programs in BASIC.

Normally, each line of text has to be typed in with a line number and then the text has to be formatted for the proper end of the 38 column screen display, and the necessary carriage return. Editing of lines and phrases becomes a real headache because it can throw the following lines out of format. If this method is not done properly, some of the words get chopped in the middle, while others end up on extra lines. It can be quite cumbersome and tedious if many lines of text are to be printed to the screen.

The following is a method which I think is quicker and easier.

1. Boot up Text Wizard and type in your text as if you were writing it for normal printout. Don't be concerned about how many words to a line, and don't put in any unneeded carriage returns.
2. Edit your text and make any changes in the usual manner.
3. When the text is satisfactory, move your text down one line, and place the following margin code on the very top line, [ctrl R]780.
4. Insert a formatted disk in your disk drive.
5. Press Option T, then Option P and type in a name for your text.
6. Your text will now be saved on the disk with your file name, and the proper carriage returns inserted.
7. Boot up DOS and select Option B for the BASIC CARTRIDGE.
8. Type ENTER "D:FILENAME", and then LIST, and your text will appear on the screen in individual lines. Each line will begin with the word ERROR, a dash and a space. Replace these with the line numbers you wish to use, the PRINT statement and Quotes. When you RUN this program the text will be displayed on the screen in perfect format and even be right justified.
9. It can now be merged with your BASIC program and you will have very neat documentation on the screen.()

PARTS FOR ATARI STARFLEET SEPT 84 (25)

B&C computervisions

** August 1984 Special **

In May 1983 Atari stopped production of the 800/810/400. On the assembly line were assembled & tested PC boards ready to build thousands of these units. B&C purchased a large quantity of these PCB's & is now selling them at very low prices (prices subject to change). These PCB's are Atari surplus items & are only available while the supply lasts. Production has stopped, demand is high, prices will ? On certain items we have quantity prices which vary depending on availability.

Happy Interface for Atari 810 $199. for Atari 1050 $225.
****** New Atari 810 Disk Drives(assembled by B&C) & Drive Parts ****
Atari used two drive mechanisms in the 810. The MPI with wide eject door & the Tandon with the narrow lift up door. We have both available.
Happy 810 disk drive in Anodized case w/cable, power pak & Happy $450
Happy 810 disk drive in Atari case w/cable, power pak & Happy $475.
Atari 810 in Black Anodized aluminum case w/cable & power pak $270.
Working 810 Side, Data Separator, Analog/Power PCB & Drive Mech $195.
Atari 810 side PCB $50. w/data sep $70. Analog/Power PCBs $55.
Complete MPI or Tandon disk drive mechanism $100. Tandon Door latch $12.
For MPI: Motor $30. Stepper $45. Head $70. Pressure pad $3. Bearing $30.
I/O cable $15. Mounting Plate $15. Black Anodized case $50. Power Pak $20.
******************** Surplus Atari Computer Parts **************
Atari 800/400 Field Service Manual (FSM) $25. SALT Diagnostic Cart $25.
Atari 800 CPU, 10k ROM, 16K RAM(no chips), Main & Power PCB $55.
CPU w/GTIA $20. 10K ROM rev B $15. 16K Ram(no chips) $3. RAM case $3.
800 Main PCB $30. 400 Main PCB $20. 16K ROM Cart PCB $3. 8K $1.
800 Power PCB $5. w/cut 10 for $10. 400 Power PCB $4. w/cut 10 for $5.
GTIA, CPU 6502, 6511, ANTIC, CTIA, PIA, POKEY, 6532, 6507, RAM 6810 $5. ea
Atari Basic PCB, Assembler PCB or Microsoft Basic PCB w/manuals $23.ea.
Atari Basic Manual, Atari Assembler Manual or Microsoft Manual $5.ea.
ROMOX 8/16K 2732/2764 EPROM cart pcb with solder splashes 10 for $20.
Cases for Atari game pcb $4. Cases for ROMOX EPROM pcb 10 for $10.
Pilot Primer or Pilot Student Manuals $5.ea. Pilot Cart $15.
*** NEW Atari 800 48K (The original, the best.) $275.****
Trak AT-D2 with printer port & TURBO software $400. Slave $295.
Percom RFD double density $395. PD w/printer port $445. Two sided + $50.
ATR 8000 or Percom compatible slave drives 1 side $225. 2 side $275.
The top rated Eclipse Top DOS $40. DOS XL $30. Atari DOS II game disk $5.
Atari 850 Interface $210. Printer cable $35. Axiom Printer Int. $99.
ATR-8000 64K w/CPM $575. 16K $425. Bit 3 80 col. $275. w/32K $300.
MPP-1000C Modem $149. Atari 1035 Modem $129. Atari Keypad $99.
Atari Touch Tablet $75. Koala Touch Tablet $85.
Axalon RAM 128K $275., 48K $95. & 32K $55. Atari 16K $25. 48K $75.
64K 600XL mod $125. Omnimon Std. $99, w/8k $140, 8K chip $45. 4K chip $30.
Printers: Star 10X $299. Prowriter 8510A $399. Super 5 CP-80 $299.
OSS Action, MAC-65 cart, Basic XL, Atari Writer, Atari Logo $85.ea
Bit Convologic Byte Writer $189. Byte Reader $35. Examiner $89. WAM $50.
Synapse & Atari games 20% off, WICO Joystick $25. Trackball $35.
*** Terms & Conditions: Prices do not incude shipping or Calif sales tax & are subject to change. All sales final. No returns or refunds. Store hours Tuesday-Friday 10:AM to 7:PM, Saturday 10:AM to 5:PM. We will ship however, we are not a mail order store & prefer walk in business. Shipping UPS COD cash, cashiers check or pre-paid. MC & Visa in store only.

3400 El Camino Real Santa Clara, CA 95051 408-554-0666

PEEKS & POKES

SLO-POKE SEP 84 (26)

DAVE GROSE, EDITOR

POKE 6,112 & 53774,112: Disables break key. (1,247)
POKE 9,255: Causes the system to lock up if the system reset key is pressed. (1)
POKE 16,64: Disables <ctrl> 1. (192)
POKE 16,64 & 53774,112: Disables the break key. (192, 247)
PEEK(18),(19),(20): Shows how many 60ths of a second have passed since the last coldstart.
For seconds:
INT((PEEK(18)*65536+PEEK(19)*256+PEEK(20))/600)
For minutes:
INT((PEEK(18)*65536+PEEK(19)*256+PEEK(20))/3600)
POKE 65,0: Shuts off the disk drive and casette I/O beeping. (3)
POKE 77,129: Starts the attract mode immediately. Poking it with 0 will stop the attract mode. (0)
POKE 82,X: Sets the left margin to X. (2)
POKE 83,X: Sets the right margin to X. (39)
PEEK(84): Current cursor row position.
PEEK(85): Current cursor column position.
PEEK(195): Holds the number of the error code. (0)
POKE 202,1: Causes the system to screw up on break or system reset.
POKE 559,0: Turns off the screen for faster I/O. (34)
POKE 559,253: Narrow playfield. (34)
POKE 559,255: Wide playfield. (34)
POKE 580,1: Causes a coldstart when system reset is pressed. (255)
POKE 694,X: Causes a key to give a different character when pressed than usual.
POKE 703,4: Adds a text window to Graphics 0. Use ?#6; to print to the top of the screen. (24)
POKE 709,X: Changes the brightness of the Graphics 0 characters. (202)
POKE 710,X: Changes the screen color of the Graphics 0 screen. (148)
POKE 712,X: Changes the Graphics 0 border or the screen on other graphics modes. (0)
POKE 752,1: Turns off the cursor. (0)
POKE 755,4: Turns characters over. (2)
POKE 756,224: Allows lowercase in Graphics 1 and 2. (224)
PEEK(764): Holds the internal code of the last key pressed. (255)
POKE 766,30: Acts as if <esc> is pressed before control keys. (0)
POKE 832,13: Presses return constantly after the program is ended. (12)
POKE 838,166 & 839,238: Sends output to the printer instead of the screen. (163,246)
POKE 1913,87: Cancels read/write verify.
PEEK(53279): Returns 3 if Option is being pressed, 5 if Select is pressed, 6 if Start is pressed, and other numbers if combinations are pressed. (7)
PEEK(53770): Returns a random number between 1 and 255.
PEEK(53775): Returns 251 if a key has been pressed or 255 if not. (255)
POKE 54018,52: Turns on the casette motor. (60)
X=USR(58464) or (61733): Does a coldstart.

What Language Does Your Atari Speak?:
A Guide to Programming Languages
Available for the Atari Computer
By Arthur Leyenberger - JACG

This is the first part of a series of articles dealing with programming languages available for the Atari computer. Over the past two years, there have been various articles in the JACG Newsletter discussing this or that language but there has been no attempt to discuss them within the context of how they compare with each other.

Many people think of the Atari computer as merely a "game" machine. Indeed, it functions superbly in this role but the uninitiated user does not realize that beneath the hood of the Atari 400/800/1200 and XL computers is a serious microprocessor that can be used with over ten different programming languages. These range from the venerable Basic (3 different dialects) to the esoteric LISP.

In an effort to give both the experienced and novice user an introduction to these capabilities, this series of articles will present an overview of the programming languages currently available for the Atari computer. The intent here is not to give you an in-depth tutorial on each of these languages, but rather to increase your awareness of the potential of your computer (I assume that you already own at least one Atari computer).

Learning a new computer language is no different than learning any other new activity. A good book on the subject will go a long way towards helping the neophyte. At the end of the series I will include a brief description of some useful books on the various computer languages. I will make an effort to indicate whether the particular book is aimed at the beginner or advanced user.

Hopefully, after reading this article, your interest will be aroused and you will have the ambition to learn a new language. Maybe the result of your efforts will someday appear in this newsletter or be a commercial product.

Machine Language

Programming a microcomputer in binary form (using the digits 0 and 1) is called machine language programming. This is the only language that can be directly understood by a microprocessor.

Early microcomputers like the Altair and Imsai used front-panel switches to represent the binary digits zero and one. A switch in the "on" position was a "one", in the off position "zero." Some switches represented instructions and others represented the memory address of the instruction. Light-Emitting Diodes (LEDs) on the front panel indicated the state of the switches. However, the process of flipping a dozen switches to manually enter the instructions and memory locations was tedious and subject to much error. Fortunately, the Atari computer is not programmed this way.

Assembly Language

Machine-language programs are almost impossible for humans to read, so symbols are used to represent the instructions. These symbols are called mnemonics (memory aids), and the 6502 microprocessor in the Atari has a unique set of them.

The process of using these mnemonics to write a program is called Assembly Language programming. The program that translates the assembly language source code into the machine language object code is called the Assembler. Some Assembler programs have the capability of creating and using a collection of routines called macros, and are therefore called Macro Assemblers. A macro is a collection of one or more statements that have been previously defined in the program which may be called by a single mnemonic. It will cause one or more machine instructions to be assembled and the binary code generated.

Although not as difficult to use and debug as machine language code, assembly language is still optimized for the machine, not the user. There are approximately six different Assembly Languages for the Atari computer (see Table 1). They vary in their complexity, flexibility, size and ease of use. If you are interested in learning how to program in Assembler, it is important that you use a good book since the language packages themselves rarely have any tutorial information. In fact, this is true for any of the programming languages mentioned in this article.

Disk Operating Systems (DOS)

A disk-operating system (DOS) is the control language that makes everything in the computer work. The DOS contains all of the system utilities used to format diskettes, copy files and entire disks, and make a back-up copy of the disk system. It also provides the input/output control for the language currently running on the computer.

The Atari 810 disk drive uses a single density disk-operating system (DOS 2.0S). The new Atari disk drive (Model 1050) is capable of running the new DOS 3.0D. This DOS increases the storage capacity of the disk drive to 127K Bytes (about 1-1/2 times the capacity of the 810). The language and the application programs that you choose all depend upon the operating system that runs on the computer. Sometimes a specific application program (like Letter Perfect by LJK) uses its own DOS. Usually, you can only run programs that use the DOS that comes with the computer. The Percom double density disk drive also has its own DOS. There is even a separate DOS called K-DOS which is especially useful to Assembly Language Programmers.

K-BYTE's DOS is a Disk Operating System for the Atari computer and is an alternative to Atari's Disk Operating System, DOS 2.0S. It offers a greater level of control over devices and memory and appeals mainly to advanced programmers.

One of the main differences between K-DOS and Atari DOS is that K-DOS is memory resident. This allows most of its features to be readily accessible, but at the expense of using more memory. In fact, when a Basic cartridge is inserted, the amount of free memory available is approximately 7K Bytes less than with Atari DOS. Part of this increased size is the due to the english language error messages that are used, rather than less memory consuming error numbers.

In addition to providing improved DOS commands, K-DOS contains a complete machine language monitor. This enables you to examine and alter memory in either HEX or ACSII format and to execute a machine language program in two ways. Also, certain DOS routines may be accessed by one-word commands.

BASIC (Beginners All-purpose Symbolic Instruction Code)

Basic is the most popular of all computer languages. It is also the most versatile. Not really one language, Basic is a family of languages having a common core. The major differences in dialects are a result primarily of the different graphic commands peculiar to the specific computer. Basic was invented in 1963 at Dartmouth College by Professors Kemeny and Kurtz to enable non-computer science students to use the school computer. A Basic program consists of statements on numbered lines which are executed one at a time. The program can be made to jump around successive statements, or to other sections of the program, and then return to execute the next line in the program. Control of the program operation is executed via a few easily learned commands, such as PRINT, GOTO, READ, and INPUT.

Basic has become popular mainly because it is so friendly. Other computer languages are complicated and use unfamiliar words, symbols, and syntax. Basic speaks a very simple English, using only a relatively small number of words that can be understood from the start.

Basic does have some drawbacks due to its inherent lack of structure. It is often said that, in Basic, programmers have too much freedom to jump around. If a complex Basic program is not well documented with comments, it is difficult for even the author to understand the program. Although Basic is simple, it must be "spoken" with precision and it will not tolerate sloppiness. There are a few ground rules that must always be followed.

There are three versions of Basic for the Atari computer. Atari Basic is the most popular and was the first version available. It exists on an 8K ROM Cartridge and is now built into (Revision B) the Atari XL computers.

Basic A+ by OSS

Atari 8K Basic (the cartridge) was originally developed by Optimized Systems Software (OSS). Basic A+ is an extension of the original language that includes over 40 new features and functions. It comes on a disk and occupies approximately 16K of memory.

Many people feel that Basic A+ is the Basic that Atari should have released initially if they had not been in such a hurry to get a product out the door. In any case, Basic A+ is easier to use than Atari 8K Basic and allows the programmer the ability to add structure to his or her programming. This is done with statements like: IF..THEN..ENDIF and WHILE..ENDWHILE. PRINT USING allows formatted output to either a printer or the screen.

Other improvements include a TAB function, an INPUT statement that allows a self-trapping prompt (it will automatically re-prompt if the input causes an error) and the ability to use subscripts with the INPUT and READ statements. Additional string handling functions are provided such as concatenation and the ability to search for a substring with the FIND function.

A couple of nice debugging functions are included. TRACE allows meaningful error messages to be displayed when testing a program. IF ERR is a function that can test error conditions and direct program flow. Also, groups of lines may be deleted at once with the DELete command.

One of the best features of Basic A+ is the extensive set of Player Missile Graphics commands. These functions make PM graphics as easy to use as PLOT and DRAWTO. In addition to these 14 commands, the joystick commands have been re-done to be easier to use and yield better movement.

OSS has recently introduced a new language called Basic/XL. This language is really an upgraded Basic A+ that comes on a bank-selectable 24K uper-Cartridge instead of a disk. Basic/XL has been covered elsewhere in previous issues of this newsletter.

See you Next Month

That's it for this month. Next month I will pick up the discussion with Atari Microsoft Basic. Then we'll talk about Pascal, C and maybe even LISP. So be sure not to miss it.

AACE 9-84

SynComm REVIEW

By Gordon Andersen

"SynComm", a communication program that is as easy to use as dialing the phone. It is menu driven with simple easy to follow keyboard entries. This system is loaded once and then put back into the package. There is also some utilities on the back of the disk that will help you set up many different types of configurations. There is also a file made up for you that sets up your terminal for full use of CompuServe. You have all the Baud Rates from 110 to 9600. Most common for home use is 300 and 1200 respectfully.

One of the features that I like best with SynComm, is the fact that you can save all or part of the text as it is sent to you. This means that as the characters are put on the screen they can be saved in blocks of memory. Those blocks of memory are as big or as small as you want them to be. Each block is set up the way you want and can be sent to disk for storage. You can print these blocks to your printer as well as saving them on disk. I don't know how many times I wanted to save something that was on the screen. Now it's as easy pressing a the right key on your keyboard.

SynComm can be set up to use the Christensen Protocol or I-modem as most people know it by. The use of this option is easy as hitting the right key to get it in operation for you. Under this I-modem system, transmitting and receiving are done with two easy key strokes. Being able to check the data as it comes to you is very important and can make the difference in if your program works or not.

More feature that might be of interest the experienced communication users. The ability to see your data being displayed in either ATASCII or ASCII. There is also the ability to display the key codes and screen editor functions. All this can be done almost any time during your use of this program. This can be done to include all incoming and outgoing translations. You also have the 8-bit ATASCII there for your use.

Here are some of the many other fine features built within this program. One, Echo Control Characters, this will let you decide if you want make your control characters visible. Second, Slow Transmission, as stated this slows down transmission to a slower bulletin board.

The ability to do all the standard public domain communication features plus the many extra features makes this program worth having on your modem software. I am making this my software to talk to CompuServe and the other bulliten boards that I access. I've used it from the day I picked it up, and put my Random disk back into it's long term storage area. I've enjoyed it from the first time I turned on my computer with this disk in it.

The documation is maybe the best I've seen for any communication program on the market. If you are going to join the computer communication explosion, then this program is for you. Having all these features at your finger tips is wonderful. The first timer as well as the season veteran will enjoy using this program.

SynCron REVIEW

By Gordon Andersen

(29)

"SynChron", a personal appointment management program (pamp). The first thing I asked myself was, what could I use a pamp program for? So I will take a penetrating look into the usefulness of this program. First do I have so many things to do that I need to keep track of my daily activities? Not really, was my first impression as I started reading the manual. The manual is of fine quality and easily followed. It was really hard to put it down. That may sound funny, but for a person who dislikes reading as much as I do, this is something.

After getting into number three of these new Synapse programs I began to get a feeling that it was going to be of the same high quality as were the others from Synapse. The manual gives you the whole story on what this program can do for you. I read the manual and then booted the program. I did run into a problem. The first time I booted, the screen did not show what it was suppose to. What to do? I went and got my sons 800 and tryed it on his system. What I found out was that my Mosaic was the problem. So I stepped through the first section of the program as if I was reading it from the screen. I was then greeted with the main menu and it's six options.

This program has a tutorial that guides you through the many features, at your speed. As for creating your daily, monthly, and yearly plans it's as easy as typing in what you are going to do that day. When you go to the yearly calendar to enter or check your entries, moving around can be done with a joystick. When a joystick is talked about, people think it is a game. This is not a game, but a great record keeping system. The use of a joystick gives it a plus. You can still use the keyboard, but using the joystick makes it easier to get around.

Birthdays and appointments are just the beginning of the list of items that you can use SynChron for. As far as, with this program I will not miss anyone of my anniversary. This in itself will make the program worth having. My wife can now keep her and the kids doctors appointment on the computer and list them out for that month. Being able to do this for two years into the future is a plus that could be very handy. There is a problem I found with this program that I'll not have worry about. That is, this program is only good until the year 30000.B.. Guess that my great, great grandson will not be able to use this program.

I see a real use for this program if you have a small business or you are a salesperson. Keeping track of these appointments are sometimes very important in making a sale. Even though I don't run a business or work as a salesperson this pamp program will be helpful. School is starting and the uses for my kids are to numerous to write down here. I feel that this is a good reason for owning this program.

Review.....

SYNFILE+

by *Greg Beaulon*, ACAOO

Nearly a year ago I decided to fill a long open gap in my program library. I needed a good data base management program (DBMS). I had several "public domain" DBMS programs, all written in BASIC, but these were all lacking in the features I desired. They were generally lacking in flexible print formatting, limited in the search/sort capabilities, and slow to the point of being painful.

After speaking with a number of friends about which DBMS is best, I decided on Filemanager 800. There was one problem with this decision: Synapse had just announced a successor to Filemanager, Synfile+. Now, I couldn't possibly go out and buy a program destined for so rapid an obsolescence, so I decided to wait for Synfile+'s release. Well, it was not a short wait. Only within the last two months has this program become generally available. But I was not disappointed.

Here's an overview of Synfile+'s features:

1. Up to 60 fields per record, for a maximum of 1000 bytes

2. Sort or search on up to 10 fields, with the operations =, /, <, and > supported.

3. Search using wild cards at any position within the field.

4. Multiple disk files supported (up to 16 disks per file).

5. Full screen editing of forms.

6. Prints in list or label formats, up to 132 characters wide.

7. Printer control characters can be imbedded in the page titles.

8. Supports single and double density disk drives, ramdisk (Axlon and Mosaic).

9. DIF function for interconversion of file formats.

10. Flexible merge and subfile functions.

11. Data can be merged with ATARIWRITER files.

Synfile+ is entirely menu driven, making it very easy to use with a minimum of referring to the manual. The menus are, at most, three levels deep and are available instantly so they have proven as easy to use as direct commands.

The routine to create a new form is actually a subprogram and is called through the open file routine of the FILES menu. After loading the routine you create a form by positioning the cursor anywhere on the 80 column (scrolled) by 24 row forms screen and type in the field name. There are 11 different kinds of fields available: text, table look-up, date, conditional, numeric, dollar, integer, computed, cumulative, record #, and counter. With a maximum of 60 fields per record I do not see me exceeding the form capacity soon.

Other features of the FILES menu allow copying, merging, subfiles renaming of files, disk formatting, density change, and use of the data interchange format (DIF). The merge capabilities have come in handy several times when I inadvertently omitted a field from a form. One peculiar feature of the drive density select is that while the 1050 "dual-density" is supported, DOS 3.0 format files are not. The DIF commands are used to swap data between Visicalc (or Syncalc) and Synfile+, a feature which I have not yet used.

The RECORDS menu is used during data entry, allowing entry, retrieval, re-indexing, deletion, and mass record updates. The data entry is displayed exactly as the form was created, with the cursor advancing a field after each entry. You may retrieve records for update selectively, either individually or all at once, which need any of the record search criteria. The index (s) by which is used to access records) may be set on up to 10 fields. A variety of information about the file is displayed during record entry, including the number of records present and capacity. The capacity is determined, for the most part, by the length of the indexed field(s) and the amount of memory you have. With a memory expansion such as a ramdisk you can have very large files (up to 16 double density disks long). I doubt if files longer than two or three disks are practical with the ATARI's disk access speed, but it is comforting to know that you are not left SOL if your file grows beyond one disk. Individual records may be printed or calculated at any time.

The REPORTS menu allows either list or label format output to the screen, printer, or disk. In the list mode you may print a columnar listing of your file (or any part of it, with conditions) up to 222 characters wide (for wide carriage printers in condensed mode). Numeric fields allow use of a total calculation simply by placing a "." after a field name. A page title line of the list mode allows embedded control codes for printer control. The label format allows printing of labels up to 80 columns by 21 rows. This format is useful for mailing labels and invoices.

While my overall opinion of Synfile+ is very favorable, the program does have its annoyances too. For one, the program must be in disk drive 1 and the disk to format and create a file page on must be in disk drive 1. This creates a lot of disk swapping when creating a new file which is one of the major reasons that I bought 2 disk drives: one for the program and one for the data. The menus, for the most part, do not have an option of over riding the default settings. Every time I print a new form, I print to the screen first. The program defaults to the printer, requiring me to mean select the screen every time. There is no provision for saving a listing format (the actual fields you desire to list), so you must type in all fields with every new listing, even if you are only changing the spacing. The tutorial (more on this later) would be much better if done in the form of a help file which can be accessed while running the program. The program does not appear to reserve much of a buffer for record input and the disk updates after every couple of modest-sized records, slowing down the record input process.

I'm sure to find more aspects of the program I like and dislike the more I use it. With a list price of $100 ($99.95, actually), it is not meant to compete with DBMS's such as dBase II (at about $400). It does not interface to programming languages, does not have multi level (or any) security, does not produce audit trails, does not perform batch processing, and does not perform a whole array of other tasks that more complex (and expensive) DBMS's are designed to handle.

About six months ago I had the opportunity of attending a market survey of Synfile+'s "tutorial disk". This was my first exposure to the program and one of my first to "tutorial" programs. I left the survey with a very favorable opinion of Synfile+ and a poor opinion of the tutorial. The tutorial is written non-interactively, meaning it is about like reading a book about someone using Synfile+; nearly worthless. The tutorial conveys too much information without offering any reinforcement that you are comprehending any of it. There is one saving grace for the tutorial: the program manual is actually pretty good.

(30)

SynStock REVIEW

By Gordon Andersen

"synStock", as you might have guess this is a stock charting program from Synapse. It's a serious stock program that takes both the newcomer as well as the expert into the world of stocks. The manual and program made it easy for me to get involved in the world of stocks. I am a newcomer to this group of people who work with stocks. I did enjoy learning about the skills needed to start setting up my very own portfolio. I had never used a program like this before. I did not know what to do or where to start. So I started from the top, reading through the manual and it's fine tutorial.

The main menu has four areas for you to use to get started. They are Download Quotes, Update Quotes, Chart Quotes and Utilities/Init.. Getting into each one would make this review to long so I'll just cover a few points of each.

First Download Quotes. This area really suprised me. I didn't know that you could download from CompuServ with this program. What can you download? Well any of the stock opening, closing and stock information that is there. Saving this data for you to use for your stock charting can make setting up the charts the way you want. You can also enter data from the newspaper using the keyboard.

Speaking of charting, this program has a graphic charting section so that you can see what your stock has done for you over a period of time. This easy to use graphing program takes all it data from within the program and lays it out in several formats. Seeing my stock charted over a few months gave me a new perspicuous about how it was doing.

Second is Update Quotes, has worded you can tell what it means, plus much more. It takes you to a sub-menu that does seven other items. These items in the sub-menu include View Portfolio, Download Entry, List Quotes, and Chart Quotes. There are three others to help you Update your stock quotes.

Third is Chart Quotes, it also takes you to a sub-menu. Within this sub-menu is four helpful routines to get you to a screen or hard copy of your stock activity. Examples of this sub-menu are Autorun, and View Portfolio. I also wondered what Autorun was. This Autorun is a way to "automatically run a sequence of commands that you define and then print price charts". The manual described it best so I used it. This is a fun feature to use and enjoy.

Last is the Utility/Init. of the main menu. As written it does just that for you. You have the options to Create DIF File, Split Adjustment, Transfer Stock, and Change Symbol. This is only four out of six subjects that you have at your command.

As I've only seen to stock programs for the ATARI, I'd have to say that this is really a good program. I have see others on different systems, and this one is in that group. To be able to see what a stock is doing can help you plan for a profitable future. I have all was looked at my company stock each day in the newspaper and wondered what it did over the pass month. Now I can collect this information and save it to disk or printer.

Bit by Bit
Printer Control with Visicalc

By Steve Honn

Reprinted from the February 1984 issue of The Frederick Atari Computer Enthusiasts Newsletter

I just recently got a printer as my Christmas present. So, I have been dragging out all sorts of programs and trying out new portions that were previously out of reach for me. I have also borrowed programs to test them with my printer. The biggest problem that I have had is printer incompatibility. It seems strange that many software developers have the odd idea that all Atari owners only buy Atari printers. Sometimes, this is true, but the best selling printers on the microcomputer market are made by Epson, NEC, C. Itoh, Gemini etc, not by Atari or Centronics. Most printers use their own (different) control codes to activate their special functions. So, to compensate, some programs go ultra utilitarian and simply use the default printer mode - Bank Street Writer is one of these. Others may include one additional printer - Letter Perfect (old version) allows use of an Epson as well as the atari printer. Visicalc for the Atari, however, supports only the Atari 825 and default printer mode.

Imagine that a program with a list price of $250 won't let me use compressed pitch with my new Prowriter printer. I was really disappointed, since I had intended to try Atari Visicalc and make some printouts for work. The sheets I needed to make had more than 80 columns, so I wanted to use the printer's compressed font. After some frustration, I opted to boot the computer with BASIC and set up the printer using LPRINT statements to send the appropriate codes to the printer. I then booted Visicalc and did my work using the compressed pitch already selected. This was a rather clumsy way to control the printer. If I needed to change any printer control settings, I had to exit Visicalc, boot up DOS and BASIC again, send the new printer control codes, then reboot Visicalc. If you have ever used Visicalc, then you can appreciate how long it takes to do all this and how it got old very quickly.

After consulting the Visicalc manual and the book Your Atari Computer, I noticed that the codes entered to select and deselect the compressed font on the Atari 825 were the ASCII character for the desired option preceded by an escape sequence. I searched the Visicalc disk code with DISKEY looking for the

Atari 825 printer escape sequences, but did not find them. It happens that the sequence to activate compressed mode on the 825 is ' ESC T' and to activate pica font requires ' ESC S'. The Visicalc manual says that entering compressed mode involves the printer setup command '/PP"CTRL T'. To leave compressed mode and reenter pica requires you type '/PP"CTRL S'.

Now, this started looking easy. The Prowriter uses the sequence 'ESC Q' to set the compressed mode. So, I naturally tried using '/PP"CTRL Q' but had no luck. I tried all sorts of combinations, with no apparent success. I even included pressing the ESC key in the sequences, but that didn't work either.

Then, something strange happened. I looked at my test printout and noticed it was in the proportional font instead of the default pica font. How did that happen? I also saw that a column header on the sheet said 'eriod 1' instead of 'Period 1' as it should have been. Aha, the lights flashed over my head as I solved the puzzle.

Visicalc apparently sense the CTRL-S and CTRL-T used for setting up the 825 and sends an escape sequence to the printer followed by the normal S or T character. This works only for the 825, though. If you have a different printer, you can beat the system with your own printer codes. MY sample that ended up in proportional mode shows why. I had ended the printer setup sequence by pressing the ESC key. The first character in the spreadsheet was a 'P' from the header 'Period 1'. This corresponds to the 'ESC P' sequence that the Prowriter needs to activate its proportional mode. After additional testing, I proved the theory held for other codes. For compressed mode, I entered '/PP"ESC Q' and it worked. I found I could select any of the print modes, including emphasized. Any escape sequence that consists of an ESC or CTRL key press followed by a single character works fine, and you don't have to risk changing the code on a very expensive program disk.

Here are some control sequences that work for the C. Itoh Prowriter or NEC 8023:

Pica enter	/PP"ESC N
Compressed enter	/PP"ESC Q
Proportional enter	/PP"ESC P
Elite enter	/PP"ESC E
Start Emphasized	/PP"ESC !
End emphasized	/PP"ESC "
Start double width	/PP"CTRL N
End Double width	/PP"CTRL O
1/6" line spacing	/PP"ESC A
1/8" line spacing	/PP"ESC B
Start underline	/PP"ESC X
End underline	/PP"ESC Y

Note: 'ESC' and 'CTRL' mean pressing these keys.

I didn't have an Epson printer to try the following sequences, but here is my best guess at some control code setups for Epson and equivalents (Gemini, Riteman, etc.):

Start compressed	/PP"CTRL O
Stop compressed	/PP"CTRL R
Start double width	/PP"CTRL N
Stop double width	/PP"CTRL T
Start underline	/PP"ESC -1(minus one)
Stop underline	/PP"ESC -0(minus zero)
1/6" line spacing	/PP"ESC 2
1/8" line spacing	/PP"ESC 0
7/72 line spacing	/PP"ESC 1
Start Italic	/PP"ESC 4
Stop Italic	/PP"ESC 5
Reset to defaults	/PP"ESC @
Start emphasized	/PP"ESC E
Stop emphasized	/PP"ESC F
Start double strike	/PP"ESC G
Stop double strike	/PP"ESC H

If you wish, you may add title lines to your Visicalc worksheets printouts. To do this, enter the Print command. When you get the prompt:

'PRINT: LOWER RIGHT, "SETUP,+,-,&

type a quotation mark (") to enter setup mode, then type in a title line. You must be careful, because these characters do not appear on the screen. After you enter your title, press the RETURN key. In order to print this line and get the printer to go to the beginning of the next line, enter a plus sign (+). This transmits a carriage return to the printer, causing it to print the tile line followed by a carriage return and a line feed.

Try these tricks and let me know if you find any others printer tricks for Visicalc.

** SPEEDING UP ATARI BASIC **

by TOM DISQUE TCC 10/84

In an attempt to increase the speed of an Atari Basic program, I attempted to tackle the slowest portions first. These portions contained divide by 256 and divide by 16 arithmetic. Because the divisors are powers of two, they allow partivularly fast solutions.

The divide by 256 occurred in a subroutine that split a number representing an internal memory address into low byte and high byte, as follows:

```
X = INT(Y/256): Z = Y-X*256
```

X would contain the high byte and Z would contain the low byte. The time necessary for execution of this sequence (excluding interpreter overhead) can be calculated using the table in De Re Atari, ch. 8 p. 8-47 (FR0 is floating point register zero and FR1 is floating point register one).

Name	Description	Basic	Time
FLDOR	Floating point load using FR0	Y	70
FLDIR	Floating point load using FR1	256	70
FDIV	FR0/FR1 Division	/	10000
FPI	Floating point to integer	INT	2400
IFP	Integer to floating point	none	1330
FSTOR	Floating point store using FR0	X=	70
FLDOR	see above	X	70
FLDIR	see above	256	70
FMUL	FR0*FR1 multiplication	*	12800
FSTOR	see above	none	70
FLDOR	see above	Y	70
FSTIR	see above	none	70
FSUB	FR0-FR1 subtraction	-	740
FSTOR	see above	Z=	70
	Total time in microseconds		27100

I replaced the above with the following Basic statements:

```
X=USR(ADR('STRING'),Y):X=PEEK(204):Y=PEEK(203)
```

WHERE 'STRING' is represented by the following.

Step	Keystrokes	Assembler	Remarks	Decimal
1.	SHFT-H	PLA	Drop no. of parameters	104
2.	SHFT-H	PLA	high byte of Y	104
3.	INV ESC CTL-E L	STA 204	save it	133, 204
4.	INV SHFT-H	PLA	low byte of Y	104
5.	INV ESC CTL-E K	STA 203	save it	133, 203
6.	INV ESC CTL .	RTS	return	96

The INV keystrokes in steps 4 and 6 were to shift from inverse back to normal characters.

The time for execution of the new Basic statements is:

Name	Description	Basic	Time
FLDOR	Floating point load using FR0	USR(	70
FPI	Floating point to integer	..Y)	2400
	'string'	none	14
FLDOR	See above	PEEK(	70
IFP	Integer to floating point	204)	1330
FSTOR	Floating point store using FR0		70
FLDOR	See above	PEEK(	70
IFP	See above	203)	1330
FSTOR	See above		70
	Total time in microseconds		5454

The new statements give an increase of 27100/5454, or a five fold increase in speed.

The divide by 16 case is similar in appearance to the divide by 256 case:

```
X = INT(Y/16): Z = Y-X*16
```

The time needed for this sequence is the same as required by the divide by 256 case. The new Basic statements are the same as before, except for the new values for 'string':

Step	Keystrokes	Assembler	Remarks	Decimal
1.	shift H	PLA	drop no. of parameters	104
2.	shift H	PLA	drop high byte	104
3.	shift H	PLA	drop low byte	104
4.	inv esc ctrl E L	STA 204	save temporarily	133,204
5.	inv) esc ctl 0	AND # 15	zap upper nybble	41, 15
6.	inv esc ctl E K	STA 203	save lower nybble	133,203
7.	% L	LDA 204	reload original byte	165,204
8.	inv J	LSR A	shift upper nybble	74
9.	J	LSR A	into lower nybbles	74
10.	J	LSR A	position	74
11.	J	LSR A		74
12.	inv esc ctl E L	STA 204	save upper nybble	133,204
13.	inv esc ctl .	RTS	return	96

The inverse keystrokes in steps 5, 8, and 13 were to shift from inverse back to normal characters. This routine separates a byte into its constituent nybbles, and gives essentially the same speed improvement as the first one.

(35)

WICK CONTROLING STRINGS

Phone: (213) 380-9513

Controlling Strings with Control Characters
by Gerry Wick

String manipulation is one of the most important and useful features of any programming language. As many of you know Atari Basic does not include some of the standard commands for string manipulation, but it is possible to simulate all of the essential commands. In this article I will demonstrate the power of Atari control characters embedded in strings. They can be used to create all kinds of screen displays that would require many more lines of code to reproduce without control characters.

In the listring that follows when you see the (ESC CONTROL/DOWN), it means to first press the ESCAPE key and then hold down the CONTROL key while pressing the key with the DOWN arrow. (ESC SHIFT/DELETE) is similar except the SHIFT key and the DELETE key are simultaneously pressed after pressing the ESC key.

Experiment with the following short program to get the idea of how to use control characters in strings.

```
10 DIM I$(100)
20 I$="THIS IS LINE ONE(ESC CONTROL/DOWN)THIS IS LINE TWO"
30 PRINT "(ESC CONTROL/CLEAR)";I$
```

When RUN, you will notice that THIS IS LINE TWO did not start at the left margin under THIS IS LINE ONE, but continued at the next colum and one line below. You might wonder why I did not use (ESC RETURN). Try it. Unfortunately it doesn't work. I found that I can simulate a RETURN with (ESC CONTROL/DOWN)(ESC SHIFT/DELETE). This will work in most situations. Replace line 20 with:

```
20 I$="THIS LINE ONE(ESC CONTROL/DOWN)(ESC SHIFT/DELETE)THIS IS LINE TWO"
```

Now the two lines should both start at the left margin.

Examine, type in and run the listring below. It demonstrates a data entry screen using a string with control characters. The string as in line 20 could fill an entire screen with one PRINT statement as in line 60. Notice line 30. It allows a string to be quickly filled with a single character, such as a dash in this case. Line 70 puts the dashes on the data entry screen in the appropriate places for data entry. And line 110 through 130 are for input. Of course this entire program is very unsophisticated with no error checking, no storage and retrieval of data, etc. However, I hope that you can use some of the ideas in your own programs. With control characters you can draw and animate figures, as well as manipulate text, in any of the graphics modes.

Listring

```
5 REM INITIALIZATION
10 DIM TEMPLATE$(100),DASH$(15),LAST$(15),FIRST$(15),PHONE$(15),LINE(3)
20 TEMPLATE$="(ESC CONTROL/DOWN)(ESC CONTROL/DOWN)(ESC TAB)TELEPHONE
   BOOK(ESC CONTROL/DOWN)(ESC CONTROL/DOWN)(ESC CONTROL/DOWN)(ESC
   SHIFT/DELETE)1.LAST NAME(ESC CONTROL/DOWN)(ESC SHIFT/DELETE)2.FIRST
   NAME(ESC CONTROL/DOWN)(ESC SHIFT/DELETE)3.PHONE NO"
30 DASH$="-":DASH$(15)="-":DASH$(2)=DASH$
40 COL=20:FOR I=1 TO 3:LINE(I)=4+I:NEXT I
50 REM DATA INPUT SCREEN
60 PRINT "(ESC CONTROL/CLEAR)";TEMPLATE$
70 FOR I=1 TO 3:POSITION COL,LINE(I):?DASH$:NEXT I
100 REM INPUT OF DATA
110 POSITION COL-1,LINE(1):INPUT LAST$
120 POSITION COL-1,LINE(2):INPUT FIRST$
130 POSITION COL-1,LINE(3):INPUT PHONE$
```

(36)

USR FUNCTION

by Ernie Rice - JACG

ATARI BASIC provides the user with a fuction known as the USR function. This allows the programmer to run an assembler program already in core (the computer's main memory). Information may be passed to the assembler code, as well as returned from the assembler routine.

Passing information to the assembler routine is valuable when you wish to use the same program to perform different tasks depending on the value supplied. The return value on the other hand may be used to detetmine the success or failure of the execution of the assembler program.

As an example, I may write an assembler program to read any sector on disk. This assembler routine will be passed the number of the sector I wish to read and return a value indicating whether or not the READ was successful. Note that without this assembler program, the user would have no way of reading any sector he or she desires.

The format of the USR function is:

```
Numeric Variable=USR(Address, Value1, Value2, ...ValueN)
```

Where the Numeric Variable is the name of a numeric variable that contains the return code from the assembler routine which starts at address (or location) "Address" in the computer's memory. Value1, Value2... ValueN are N different values to be supplied to the assembler routine as input. These must all be positive integer values. Numbers such as -3, 1.25 are invalid.

So if I code the following USR statement in a BASIC program

```
I=USR(1536,1,2,3)
```

The result would be to cause BASIC to branch to the assembler code located at address 1536 (decimal), and make available as input the values 1,2,3. The return code from the assembler routine will be available to the BASIC program via the variable I.

Note that even though these input and output variables are available, it is still up to the programmer to READ the input variables in the assembler routine and also to store the value in the proper locations for the return code. The assembler or BASIC will not do this on its own.

The locations for the return code are D4, D5 (hexidecimal) or 212, 213 (decimal). Therefore in order for the BASIC program to receive the proper value for the return code, it must be stored in these locations just prior to exiting the assembler routine. The input values are placed on the system's stack.

Why do you need 2 locations for the return code? Well - the ATARI is driven by a 6502B processor. This is an 8 bit processor. Each byte is comprised of 8 bits, with each bit representing a power of 2. Therefore the maximum value able to be stored in one byte raised to the 8th power minus 1.

Format of an 8 bit processor

```
 7 6 5 4 3 2 1 0

128 64 32 16 8 4 2 1
```

Each bit can be in one of two states: ON or OFF (just like a light switch). Numbers stored in this format are called binary numbers. An ON state is indicated by a 1 and an OFF state is indicated by a 0. The value of the number is determined by adding all of the values of the positions containing a 1 together.

For example:

```
128 64 32 16 8 4 2 1

 0  0  0  0  0 0 0 0 =0
 0  0  0  0  0 0 0 1 =1
 0  0  0  0  0 0 1 0 =2
 0  0  0  0  0 0 1 1 =3  = 2+1
 0  0  0  0  0 1 0 1 =5  = 4+1
 0  0  0  0  1 1 1 1 =15 = 8+4+2+1
 0  1  0  0  0 0 0 0 =128
 1  1  1  1  1 1 1 1 =255= 128+64+32+16+8+4+2+1
```

Since the 6502B is an eight bit processor, each byte can hold a maximum value of 255 (11111111 in binary). This is not adequate for large numbers, so ATARI set aside two locations D4 and D5 (Hex) 212, 213 (Decimal). The ATARI will take the value stored in the highest location (D5/213) and multiply it by the value of 256, then add the value contained in the lower locations (D4/212) to the result. This allows for a maximum value of (256*256)+256= 65535. We now are able to pass back a value from 0 to 65535 inclusive. This provides the user a much greater range of possible return code.

The same principle is used in the passing of input variables or values as well as in the addressing scheme of the ATARI itself. By the way, did you ever wonder why the ATARI can only address 64K? This is why: Divide 65536/1024 (1024 bytes=1K). You get 64! Since the ATARI uses two bytes to address, it can only access a maximum of 65536 bytes (one extra for a displacement of zero).

You may be wondering why you are being tormented with these binary, hexidecimal and decimal tidbits? What value is it to me the novice programmer? The answer is simple. BASIC is great and easy to use, but it cannot allow all of the things that assembly can.

Sept

(37)

VERTICAL BLANK INTERRUPTS

by

Harjit Singh, ACAOC

Interrupts, what are they? They do exactly that, interrupt the machine (rather the 6502 in the Atari) and have it execute a user written machine language program in the memory whenever the interrupt it to take place and then resume whatever it was doing as if nothing happened. There are several types of interrupts built into the Atari e.g. Display List Interrupt, I will refer to them as DLI'S Vertical blank interrupts (VBI'S) Keyboard Interrupts to name a few. Most of them are transparent to the user i.e. the user is not neccessarily aware that the interrupt is taking place as a matter of fact the program that the computer was executing before it went to the interrupt does not neccessarily know that an interrupt was executed.

The next line which is exactly underneath the one it just drew. It repeats this 191 times then it is done with that "picture". At this time the beam is at the bottom right corner. The beam is then acted to come to the top left position and while it is returning to the top left position, it is turned off so it does not draw on the screen. To try and see what I mean, place a fan in front of the screen and turn them both on. If you look at it you might be able to see the screen "fall apart" or rather the screen being drawn due to an optical illusion.

Well getting back to VBI's, they are executed during the time the beam of the T.V. is going from the bottom right corner to the top left corner. So what this means is that a VBI will be executed sixty times a second. You may think what would anybody want to have something running that fast or often! The VBI comes in handy when you are doing sound and when the computer is doing "house keeping chores" e.g. incrementing the internal clock, generating random numbers, copying shadow registers etc. There are two types of VBI's, one called immediate is the one that is executed before the computer does its "house keeping chores" and second one is called deffered it is executed after the "chores".

Now that I have tried to define a VBI I will tell you how to implement it.

1. Load the Accumulator with a 6X if you want an immediate VBI or a seven if it is a deffered VBI. A deffered VBI is reccomended.

2. Load the Y-register with the low byte of the address and the X-register with the high byte.

3. Jump to a OS(Operating System) routine called SETVBV at $E45C which stores the VBI address in $222 and $223 for immediate or $224 and $225 for deffered. This is done to make sure the address is put in before the computer jumps through it i.e. if the computer puts in the low byte but has not yet put in the high byte now the computer will jump to an incorrect address since the high byte is wrong.

4. The program you want executed by the VBI should be terminated with a JMP $E46F if it is an immediate VBI or JMP $E462 if it is a deffered VBI.

Now within a 60th of a second your program should start executing. The program should not be long I find a program longer than fifty bytes starts of "hang" the system or slow it down.

I am presenting an example of a VBI that alters the speed of the cursor only. I hope to write about DLI's next month.

```
10 *= $600      ;Start adress
15 PLA
20 LDY #$E256   ;Get Lo-Byte of routine
30 LDX #$/256   ;Get Hi-Byte of routine
40 LDA #$6      ;Set vertical blank interrupt to immediate
50 ISR $E45C    ;Goto Set-Vertical blank routine
60 RTS          ;Return to Basic
70 S LDA $2F3   ;Check to see what character was pressed
80 CMP #134     ;Was it move the cursor one back
90 BEQ 2        ;If it was go to the timing routine
100 CMP #135    ;Was it move the cursor one forward
110 BEQ 2       ;If so, goto timing routine
120 CMP #142    ;Was it move the cursor one up
130 BEQ 2       ;If so, goto timing routine
140 CMP #143    ;Was it move the cursor one down
150 BEQ 2       ;If so, goto timing routine
160 JMP $E45F   ;Since none of the designated keys
165 ;Was pressed, exit the vertical blank routine.
170 Z LDA $22B  ;Get value in auto-repeat count down
175 ;register.
180 CMP #5      ;Time for the character to repeat is
190 BCC END     ;If it is less than 5 then goto exit
200 LDA #$6     ;Since it is greater than five
210 STA $22B    ;Change it to five.
220 END JMP $E45F ;Exit Vertical Blank
```

The basic version is presented here.

```
10 I=1536
20 ? "CHECKING DATA"
25 READ A
30 B=PEEK(184)*256+PEEK(183)
40 IF A=256 THEN READ A:IF A=CKSM THEN ? "NO
   ERRORS" :I=USR(1536):END
50 IF A>256 THEN 70
60 CKSM=CKSM+A:POKE I,A:I=I+1:GOTO 25
70 IF A<>CKSM THEN PRINT "ERROR IN LINE ";B:END
80 CKSM=0:GOTO 25
1000 DATA 104,160,264
1010 DATA 11,182,0,169,6,32,92,228,706
1020 DATA 96,173,242,2,201,134,240,15,1103
1030 DATA 201,135,240,11,201,142,240,7,1177
1040 DATA 201,143,240,3,76,95,228,173,1159
1050 DATA 43,2,201,5,144,5,169,5,574
1060 DATA 141,43,2,76,95,228,256,586
```

"GEEZ! TALK ABOUT A PERSONAL COMPUTER!"

V

TECHNICAL TIPS

by

Paul Surowiec

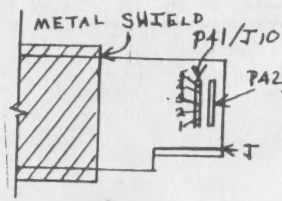
CSQUAD 9-84

There are several good reasons for installing either an all purpose write enable - disable switch or a simple write enable switch in your disk drive. With a write enable switch you will be able to write to the back of a disk without punching the disk jacket or write to a disk without a write protect notch on either side. A write disable (protect) will keep you from accidentally writing to or formatting a disk and destroying the program or data on it. These circuits are specifically for the Atari 810 drive but the same circuits can be used in most other models.

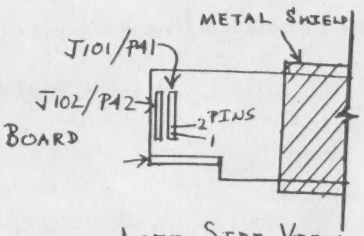
Parts required include a 150 ohm 1/4 watt resistor six to ten inches of wire (more if you plan to mount the switch on the front of the case), and a DPDT (double pole - double throw) center off switch.

To install the switch, first open the disk drive case. The screws are located beneath the four plastic stick on tabs on the top cover of the drive. Next locate plug P41 on J101. From the front, it is located in the left rear corner on the side board (see accompanying diagrams - fig. 1 and 2). Cut the red and black wires from pins 1 and 2 (bottom two wires) about an inch from plug P41. Solder the two wires coming from plug P41 to the center two pins on the switch (see fig.3). Solder the two wires coming from the sensor (or switch) on the drive mechanism to the pins on one side of the switch (fig.3). Now solder the resistor across the pins on the other side of the switch (fig.3). Tape your work. Mount the switch to the case and try your drive before you put the top cover back on. Configuing the switch to one side is write enable, to the center is write disable, and to the other side is normal operation.

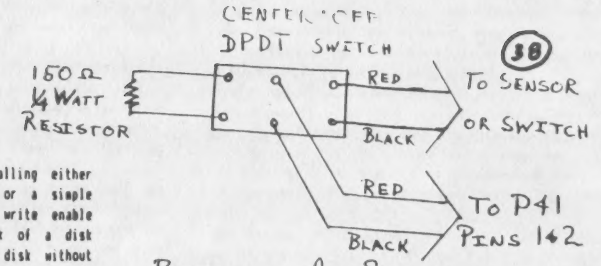
A simple write enable switch can be made by putting an SPST (single pole - single throw) switch and a 150 ohm resistor in series accross J101 pins 1 and 2 (fig.4 and fig.5) on the side board. Refer to the previous paragraph and fig.4 for locations of J101 (P41). Pins 1 and 2 are the bottom two pins. Solder the leads to the noncomponent side of the board.



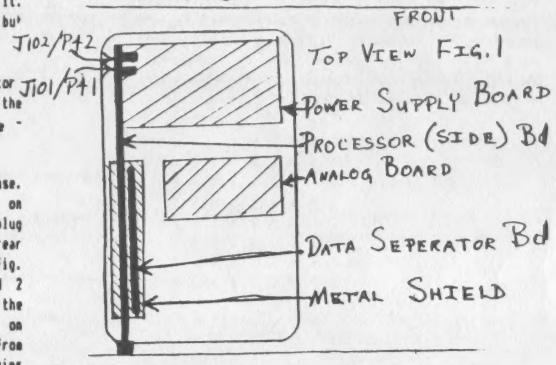
RIGHT SIDE VIEW
FIG. 2



LEFT SIDE VIEW
FIG. 4



BOTTOM of SWITCH
FIG. 3



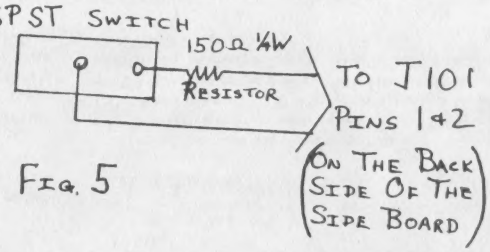


FIG. 5